

**WYDZIAŁ
ELEKTROTECHNIKI
I INFORMATYKI**
POLITECHNIKI RZESZOWSKIEJ

Katedra Podstaw Elektroniki

Szymon Starzak

Mobilny program edukacyjny

Praca dyplomowa inżynierska

Opiekun pracy:
dr inż. Maciej Kusy

Rzeszów 2015

Podziękowanie

Składam serdeczne podziękowania,
wspierającej rodzinie oraz
dr inż. Maciejowi Kusemu
za poświęcony czas i pomoc
w przygotowaniu niniejszej pracy.

Spis Treści

Wstęp	7
Rozdział 1. Charakterystyka porównawcza	9
1.1 Quizwanie	9
1.2 Kujon.....	11
1.3 Pierwiastki.....	14
1.4 Fizyka na 5	15
Rozdział 2. Szczegółowy opis aplikacji Mobilny Program Edukacyjny	18
2.1 Moduł 1: Główny ekran aplikacji	18
2.2 Moduł 2: Programy przedmiotów językowych.....	19
2.2.1 Moduł Ortografia dla najlepszych.....	19
2.2.2 Moduł Wisielec z Polski	21
2.2.3 Moduł Wisielec z Anglii	24
2.2.4 Moduł Nauka angielskiego	25
2.3 Moduł 3: Programy przedmiotów ścisłych	28
2.3.1 Moduł Szybkie operacje.....	29
2.3.2 Moduł Tablica Mendelejewa.....	31
2.3.3 Moduł Tablica Mendelejewa ENG	33
2.3.4 Moduł Systemy liczbowe.....	35
2.4 Moduł 4: Statystyki	37
2.5 Moduł 5: Koniec	38
Rozdział 3: Wykorzystane technologie programistyczne.....	39
3.1 Środowisko programistyczne	39
3.2 Języki programowania	39
3.2.1 Język Java	39
3.2.2 Język XML.....	40
3.3 Baza danych	41
3.4 Podstawowe pojęcia związane z programowaniem aplikacji na platformę Android	41
3.4.1 Aktywność	41
3.4.2 Cykl życia aktywności	41
3.4.3 Widoki.....	42
3.4.4 Kontekst aplikacji	43
3.4.5 Zamiary	43
3.5 Implementacje modułów	43

3.5.1 Implementacja modułu Główny ekran aplikacji.....	43
3.5.2 Implementacja modułu Programy przedmiotów językowych	44
3.5.3 Implementacja modułu Programy przedmiotów ścisłych	47
3.5.4 Implementacja modułu Statystyki	47
3.5.5 Implementacja modułu Koniec.....	48
3.5.6 Implementacja modułu Ortografia dla najlepszych.....	48
3.5.7 Implementacja modułu Wisielec z Polski	51
3.5.8 Implementacja modułu Wisielec z Anglii	54
3.5.9 Implementacja modułu Nauka angielskiego.....	55
3.5.10 Implementacja modułu Szybkie operacje.....	57
3.5.11 Implementacja modułu Tablica Mendelejewa.....	59
3.5.12 Implementacja modułu Tablica Mendelejewa ENG	61
3.5.13 Implementacja modułu Systemy liczbowe	61
Wnioski	64
Literatura	66
Wykaz załączników.....	67
Streszczenie	68

Wstęp

Aplikacje edukacyjne cieszą się popularnością nie tylko w szkołach i uczelniach, ale także w gronach wielu zainteresowanych użytkowników, korzystających z nich w celu poszerzenia swojej wiedzy z różnych dziedzin naukowych, bądź też dla samej rozrywki. Obecny rozwój technologiczny oraz powszechna dostępności najnowszych urządzeń mobilnych sprawiły, że wiele popularnych aplikacji komputerowych doczekało się swojej implementacji na telefony, tablety oraz na inne urządzenia mobilne. Obecnie na rynku znajduje się kilka aplikacji o tematyce związanej z edukacją. Jedną z nich jest aplikacja Quizwanie [1] stworzona w języku Java na platformę Android, która ze względu na swoją popularność doczekała się także implementacji na Windows Phone oraz iOS. Główną zaletą tej aplikacji jest sposób rozgrywki wzorowany na teleturniejach i rywalizacji. Konkurentem może być zarówno przypadkowa osoba jak i znajomy. Mocną stroną tej aplikacji jest też to, że dzięki opcji tworzenia własnych pytań, użytkownicy mogą, rozwijać bazę pytań, dzięki czemu rozrywka staje się ciekawsza, a powtarzalność pytań coraz mniejsza. Kolejną podobną aplikacją jest Kujon [2]. Przeznaczona jest głównie dla uczniów szkoły podstawowej. Jej funkcjonalność oferuje naukę w zakresie języka polskiego (ortografię i fonetykę), oraz matematyki (podstawowe działania arytmetyczne). Dodatkową opcją tej aplikacji jest tworzenie planu lekcji. Innym przykładem programu edukacyjnego jest stworzona na platformę Windows Phone aplikacja Pierwiastki wydana przez elesoft [3]. Jej głównymi odbiorcami są uczniowie szkół średnich o profilach chemicznych, ale także wszyscy którzy potrzebują informacji o wybranym pierwiastku chemicznym. Dzięki przyjemnej dla oka szacie graficznej, która przedstawia w przejrzysty sposób układ okresowy pierwiastków chemicznych wraz z informacjami o każdym z nich, program cieszy się dobrą opinią grona swoich użytkowników. Ostatnią omawianą mobilną aplikacją jest Fizyka na 5 [4]. Łatwa w obsłudze, darmowa aplikacja, która zawiera większość fizycznych wzorów wraz z opisami oraz ilustracjami obrazującymi zjawiska fizyczne. Jest idealna dla uczniów oraz studentów fizyki.

Głównym celem pracy jest opracowanie mobilnej aplikacji edukacyjnej w języku Java na platformę Android, dzięki której użytkownik będzie mógł poszerzać swoją wiedzę z różnych dziedzin naukowych. Aplikacja ta zawierać będzie zestaw modułów edukacyjnych, a każdy z nich będzie przynależeć do jednej z dwóch kategorii tematycznych: język bądź też do przedmiotów ścisłych. Każdy moduł edukacyjny

będzie interaktywną grą, w której użytkownik będzie musiał wybrać poprawną odpowiedź. W zależności od poprawności wybranej odpowiedzi, użytkownik otrzyma stosowny komunikat tekstowy lub komunikat animowany o tym, czy odpowiedział poprawnie, a jeśli nie, to która odpowiedź była poprawna. Użytkownik z poziomu głównego menu, będzie miał również dostęp do statystyk. Statystyki przedstawiać będą nazwę modułu, liczbę poprawnych i niepoprawnych odpowiedzi, a także procentowy stosunek udzielonych odpowiedzi do rozegranych partii.

Głównym elementem nowości wprowadzonym do pracy w stosunku do podobnych programów na rynku jest wielotematyczność modułów edukacyjnych. Moduły te dotyczą zagadnień języka polskiego, języka angielskiego, matematyki, chemii oraz informatyki.

Struktura pracy jest następująca. W Rozdziale 1 dokonano szczegółowej charakterystyki porównawczej podobnych aplikacji dostępnych na dzień dzisiejszy na rynku. Rozdział 2 przedstawia szczegółowy opis aplikacji. W Rozdziale 3 przedstawiono technologie użyte przy tworzeniu aplikacji. Zaprezentowano wybrane fragmenty kodu źródłowego najważniejszych modułów programu, zilustrowano schematy tabel bazy danych. Rozdział ostatni podsumowuje pracę.

Rozdział 1. Charakterystyka porównawcza

W niniejszym rozdziale szczegółowo opisano wybrane aplikacje edukacyjne, dostępne na dzień dzisiejszy na rynku urządzeń mobilnych. Należą do nich wyszczególnione we wstępie pracy Quizwanie, Kujon, Pierwiastki oraz Fizyka na 5.

1.1 Quizwanie

Jest wieloplatformową mobilną aplikacją edukacyjną, której koncepcja rozgrywki wzorowana jest na teleturniejach typu quiz. Aplikacja ta dostarcza użytkownikom rozrywki intelektualnej, polegającej na przedstawieniu przez jednego z graczy pytań z wybranej kategorii tematycznej w taki sposób, by na podstawie dostarczonych wraz z pytaniem informacji oraz wiedzy ogólnej drugi gracz był w stanie wydedukować prawidłową odpowiedź. Aplikacja oparta jest na licencji Adware z możliwością wykupienia konta Premium.

Dzięki swojej popularności, aplikacja doczekała się implementacji na platformę Android, Windows Phone, oraz iOS. Przykładowe główne okno aplikacji ukazuje Rysunek 1.1



Rysunek 1.1 Przykładowe główne okno aplikacji.

Użytkownik programu Quizwanie może jednocześnie prowadzić wiele rozgrywek z różnymi przeciwnikami. Każda rozgrywka składa się z sześciu rund podzielonych na kategorie tematyczne. Gracz, na którego przypada runda, wybiera jedną z trzech wylosowanych kategorii tematycznych (Rysunek 1.2). Następnie jako pierwszy odpowiada na trzy kolejne pytania z wcześniej wybranej kategorii. Przykładowe kategorie ukazano na Rysunku 1.3. W skład pojedynczego pytania wchodzi: treść pytania, oraz cztery możliwe odpowiedzi, z których tylko jedna jest poprawna. Na Rysunku 1.4 pokazano przykładowe pytanie. Gracz wskazuje na wybraną odpowiedź poprzez kliknięcie na odpowiedni przycisk. W zależności od poprawności udzielonej odpowiedzi użytkownikowi zostaje podana odpowiednia informacja. Jeżeli odpowiedź jest poprawna, wówczas wybrany przycisk zapala się na kolor zielony. Natomiast jeśli gracz błędnie wybrał odpowiedź, wówczas wybrany przycisk zapala się na kolor czerwony, a przycisk pod którą znajdowała się poprawna odpowiedź na kolor zielony. Każde pytanie jest ograniczone, czasem dziesięciu sekund. Jeśli gracz nie wybierze żadnej odpowiedzi w podanym czasie, wówczas odpowiedź zostanie uznana jako niepoprawna. Gracz, na którego nie przypada runda, czeka aż jego przeciwnik skończy odpowiadać na pytania. Następnie odpowiada on na dokładnie te same pytania, na takich samych zasadach jak jego przeciwnik. Jedyną różnicą jest to, że po udzieleniu odpowiedzi na pojedyncze pytanie, bez względu na to czy odpowiedział poprawnie czy nie, pokazana zostaje informacja którą odpowiedź wybrał jego przeciwnik. Po rozegraniu wszystkich sześciu rund, obu graczom zostaje pokazana informacja o wyniku. Decyduje o tym ilość poprawnie udzielonych odpowiedzi.

Aplikacja Quizwanie zdobyła swoją popularność dzięki wprowadzonemu systemowi doboru przeciwników. Liczba użytkowników tej aplikacji jest tak duża (około 150 tys.), że nawet przy wyborze rozgrywki z losowym przeciwnikiem, w ciągu kilkunastu sekund przydzielony zostanie partner do rozgrywki. Kolejną zaletą jest to, że użytkownik który wykupił konto Premium dostępne w aplikacji ma możliwość tworzenia własnych pytań, dzięki czemu przyczynia się do rozwoju aplikacji. Twórcy aplikacji mogą zaś skupić się na eliminacji potencjalnych błędów, czy dodawaniu nowej funkcjonalności. Mimo tego że aplikacja Quizwanie stanowi dla graczy świetną rozrywkę intelektualną, to nie poszerza w dużym stopniu wiedzy jej użytkowników. Wynika to z tego, że zabawa polega na dyskusji zgadujących nad różnymi możliwościami i wykluczaniu tych, które nie spełniają warunków zadanych w pytaniu.

Wadą jest też to, że do korzystania z aplikacji wymagane jest połączenie z siecią Internet.



Rysunek 1.2 Wybór kategorii.



Rysunek 1.3 Informacje o rozgrywce.



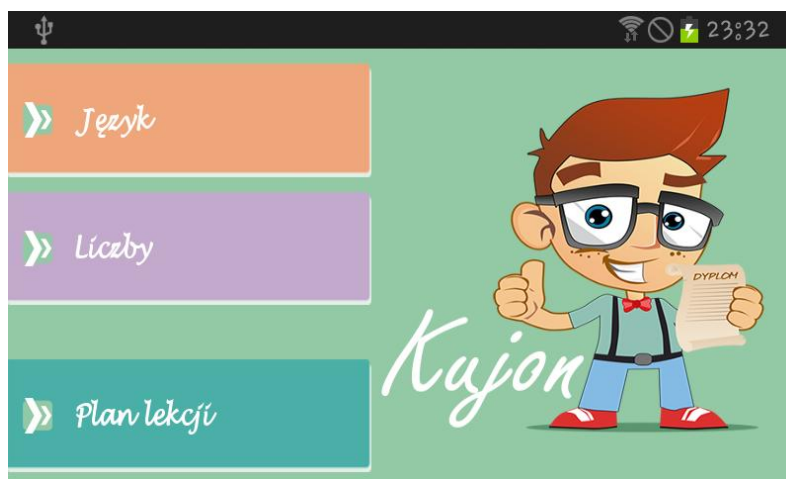
Rysunek 1.4 Przykładowe pytanie.

1.2 Kujon

Aplikacja Kujon została zaprojektowana z myślą o najmłodszych użytkownikach urządzeń mobilnych. W jej założeniach jest ułatwienie nauki w zakresie języka polskiego: zasad ortografii, fonetyki, oraz w zakresie matematyki: nauka dodawania, odejmowania, mnożenia i dzielenia w czterech zakresach: do 25, 50, 75, 100. Jest to prosta darmowa aplikacja stworzona na platformę Android.

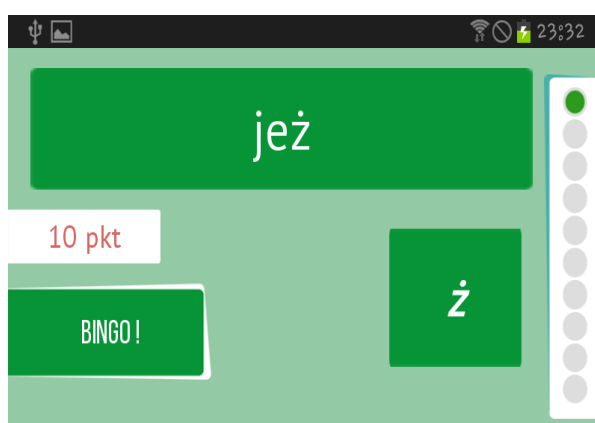
Po uruchomieniu aplikacji Kujon użytkownik może wybrać jedną z trzech dostępnych opcji: Język, Liczby bądź Plan lekcji. Rysunek 1.5 przedstawia główne okno aplikacji. Dwie pierwsze opcje mają charakter edukacyjny. Po wybraniu pierwszej z nich (opcja Język), użytkownik przechodzi do kategorii językowej. Druga opcja przenosi go do działu z zagadnieniami związanymi z arytmetyką liczb. Trzecia opcja ma charakter organizacyjny. Dzięki niej użytkownik może uzupełniać swój plan zajęć.

Podobnie jak aplikacja Quizwanie, aplikacja Kujon w swoim zamyśle bazuje na elemencie gry. Zarówno kategoria językowa, jak i ta związana z operacjami na liczbach zawiera zestawy testów, w których użytkownik ma do wyboru dwie lub więcej (w zależności od zagadnienia) możliwych odpowiedzi, przy czym istnieje tylko jedna poprawna.



Rysunek 1.5 Główne okno aplikacji.

W kategorii Język istnieje wybór dwóch zagadnień: ortografia oraz fonetyka. Jeżeli użytkownik wybierze ortografię, wówczas będzie miał do wyboru parę polskich znaków ortograficznych: 'ch' i 'h', 'ó' i 'u', 'rz' i 'ż'. Po wybraniu interesującej go pary znaków, użytkownik przechodzi do testu. Test składa się z 10 pytań. Pojedyncze pytanie składa się ze słowa języka polskiego, w którym w miejsce znaku ortograficznego wstawiono trzy kropki. Zadaniem gracza jest wskazanie z spośród dwóch możliwych odpowiedzi (par znaków ortograficznych) tej właściwej. Gdy gracz wskaże odpowiedź, wówczas otrzymuje on informację o poprawności zaznaczonej odpowiedzi. Rysunek 1.6 i Rysunek 1.7 przedstawiają przykład pytań dla testu ortograficznego.

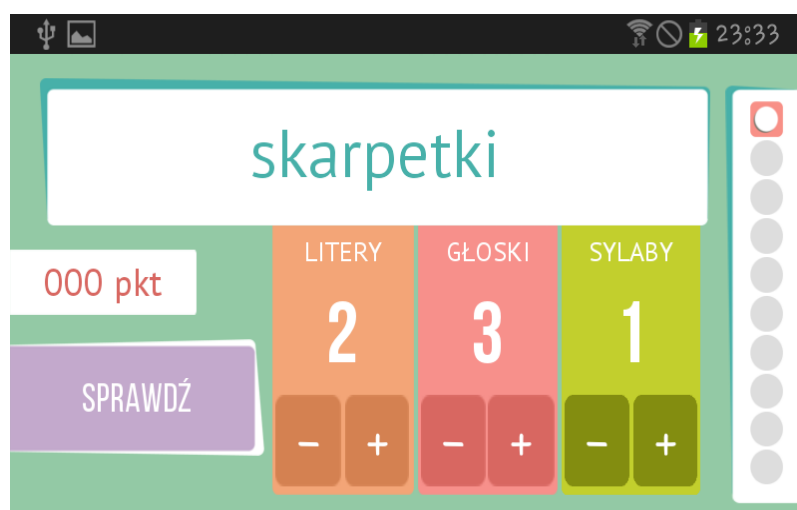


Rysunek 1.6 Przykład poprawnej odpowiedzi dla ortografii.



Rysunek 1.7 Przykład niepoprawnej odpowiedzi dla ortografii.

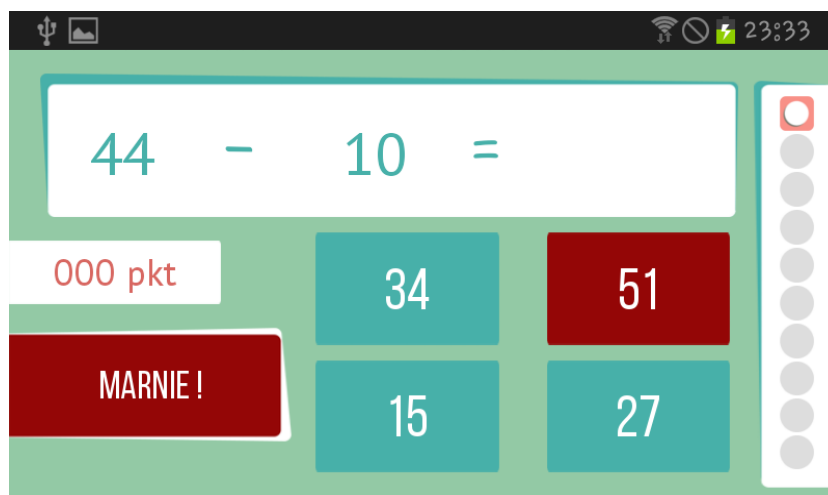
Kategoria fonetyka dzieli się na dwie podkategorie: Sylaby i głoski oraz Samogłoski i spółgłoski. W obu przypadkach, użytkownik po rozpoczęciu testu ma za zadanie zliczać w podanym wyrazie ilość wystąpienia sylab i głosek bądź samogłosek i spółgłosek. Po zatwierdzeniu odpowiedzi, użytkownik otrzymuje informacje o poprawności wybranej odpowiedzi. Rysunek 1.8 przedstawia przykładowe pytanie z testu fonetycznego.



Rysunek 1.8 Przykładowe pytanie z fonetyki.

W kategorii liczby mamy do wyboru cztery podkategorie: Dodawanie, Odejmowanie, Mnożenie oraz Dzielenie. W każdej z tych kategorii przed rozpoczęciem testu wybierany jest zakres liczbowy, którego będzie dotyczył test. Zasada działania wszystkich czterech kategorii jest identyczna, z tą różnicą, że jest inny operator arytmetyczny. Użytkownikowi zostaje podane wyrażenie matematyczne, a na jego podstawie ma wybrać jedną z czterech odpowiedzi tak, aby spełnione było równanie. Po zatwierdzeniu odpowiedzi, użytkownik otrzymuje informacje o poprawności wybranej odpowiedzi. Rysunek 1.9 przedstawia przykładowe pytanie z kategorii liczby.

Zaletą aplikacji Kujon jest to, że do korzystania z niej nie jest wymagane połączenie z siecią Internet. Wadą natomiast jest, że ze względu na trywialność zagadnień, jej odbiorcami będą tylko najmłodszy użytkownicy urządzeń mobilnych.



Rysunek 1.9 Przedstawia pytanie z kategorii odejmowanie.

1.3 Pierwiastki

Aplikacja Pierwiastki jest darmową mobilną aplikacją edukacyjną wydana przez firmę elesoft. W jej założeniach jest pomoc w nauce pierwiastków chemicznych. W przeciwieństwie do aplikacji Quizwanie oraz Kujon, aplikacja Pierwiastki nie posiada elementu gry.

Aplikacja wydana została wyłącznie na platformę Windows Phone. Można ją zainstalować z poziomu sklepu Store. Jedynym wymaganiem do pobrania aplikacji Pierwiastki jest posiadanie darmowego konta w Windows Phone.

Po uruchomieniu aplikacji, na głównym ekranie zostaje wyświetlony układ okresowy pierwiastków, wraz z podstawowymi informacjami o każdym z nich, tj. nazwa pierwiastka, symbol, liczba atomowa oraz masa atomowa (Rysunek 1.10). Rysunek 1.11 przedstawia, że w zależności od stanu materii pierwiastka i jego charakteru chemicznego, każdy pierwiastek na tablicy ma przydzielony odpowiedni kolor tła oraz kolor symbolu. Po kliknięciu na dowolny pierwiastek użytkownik ma dostęp do szczegółowych informacji na temat interesującego go pierwiastka. Informacje te to m.in. temperatura wrzenia, gęstość, wartości tlenków, wartościowość. Rysunek 1.12. przedstawia szczegółowe informacje odnośnie wybranego pierwiastka.

Pierwiastki chemiczne

1 I A

1,00794
1 H
Wodór

2 II A

6,941
3 Li
Lit

9,012182
4 Be
Beryl

22,989770
11 Na
Sód

24,3050
12 Mg
Magnez

39,0983
19 K
Potas

40,078
20 Ca
Wapń

44,955910
21 Sc
Skand

47,88
22 Ti
Tytan

Pierwiastki chemiczne

Metale alkaliczne
Metale ziem alkalicznych
Metale przejściowe
Metale grup głównych
Półmetale
Niemetale

Gazy szlachetne
Halogeny
Lantanowce
Aktynowce
Właściwości niez

Ciało stałe Gaz Ciecz Niezn

8 VIII B

55,8457
26 Fe
Żelazo

9 VIII B

58,933200
27 Co
Kobalt

10 VIII B

58,6934
28 Ni
Nikiel

11 I B

63,546
29 Cu
Miedź

Rysunek 1.10 Główne okno ekranu aplikacji.

Rysunek 1.11 Legenda do pierwiastków na głównym ekranie

liczba atomowa masa atomowa

17 grupa 5 okres p blok

$\pm I, III, V, VII$
wartościowość

fluorowec
właściwości metaliczne

4940 kg/m³
gęstość

184,25 °C
temperatura wrzenia

113,7 °C
temperatura topnienia

silnie kwasowe
właściwości tlenków

[Kr]4d¹⁰5s²5p⁵
konfiguracja elektronowa

eliesot.pl

Rysunek 1.12 Szczegółowe informacje dotyczące pierwiastka

Dzięki łatwości w obsłudze aplikacji, użytkownik ma szybki dostęp do najważniejszych informacji na temat interesującego go pierwiastka chemicznego. Zaletą aplikacji Pierwiastki jest także to, że ciągle jest ona rozwijana i z każdą nową aktualizacją dodawane są nowe informacje o każdym z pierwiastków chemicznych. W przeciwieństwie do aplikacji Kujon, aplikacja Pierwiastki stworzona jest dla szerszego grona użytkowników: uczniów, studentów, a także dla tych, którzy na co dzień zajmują się chemią. Główną wadą aplikacji Pierwiastki jest brak elementu gry. Użytkownik korzysta z niej jako z pewnej formy ściągawki, gdy potrzebuje informacji o pierwiastku chemicznym.

1.4 Fizyka na 5

Kolejną mobilną aplikacją edukacyjną jest aplikacja Fizyka na 5. Stworzona przez firmę Geckonization [5], na licencji Adware. Jest łatwą w obsłudze aplikacją, która zawiera większość fizycznych wzorów wraz z opisami oraz ilustracjami zjawisk fizycznych. Na dzień dzisiejszy aplikacja zawiera następujące rozdziały dotyczące zagadnień z fizyki:

- Działania na wektorach
- Ruch prostoliniowy
- Ruch jednostajnie przyspieszony
- Rzut ukośny

- Siła
- Praca, moc, energia
- Ruch obrotowy
- Ruch harmoniczny
- Grawitacja
- Fale poprzeczne i podłużne
- Fale dźwiękowe
- Sprężystość
- Elektrostatyka
- Pole Magnetyczne
- Prąd stały
- Prąd zmienny
- Optyka falowa
- Fale elektromagnetyczne

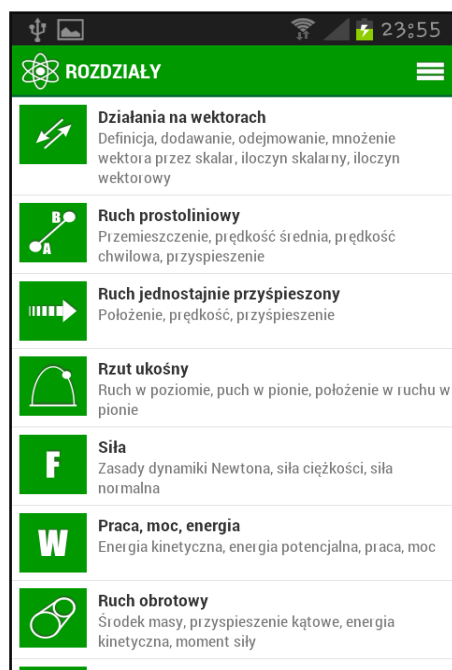
Aplikacja dostępna jest wyłącznie na platformę Android. Można ją pobrać w Sklepie Google Play. Nie wymaga ona od użytkownika zgodny na dostęp do żadnych specjalnych systemowych usług urządzenia.

Po uruchomieniu aplikacji użytkownik ma do wyboru z listy jeden z rozdziałów zagadnień fizyki. Fragment listy dostępnych rozdziałów przedstawia to Rysunek 1.13.

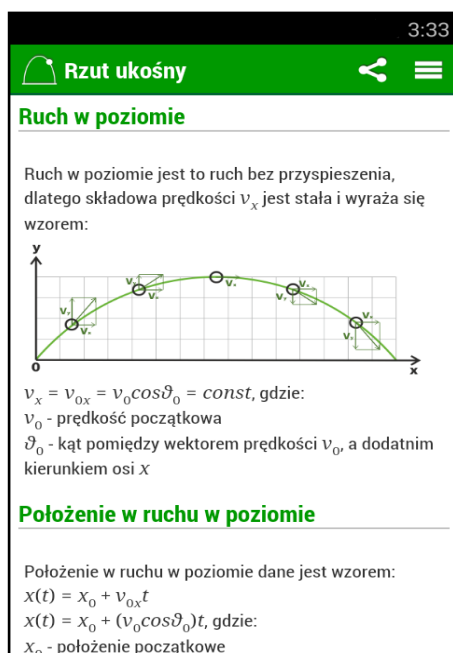
Po wybraniu interesującego użytkownika działu, na ekranie użytkownika ukazuje się okno, na którym użytkownik wyświetlone ma wszystkie informacji na temat interesującego go zagadnienia fizycznego, przedstawionego pod postacią definicji, wzorów oraz ilustracji. Rysunek 1.14 i Rysunek 1.15 przedstawiają przykładowe działy. Do głównych zalet aplikacji Fizyka na 5 należą:

- Prosta i piękna szata graficzna.
- Aplikacja jest stale rozwijana, dochodzą nowe zagadnienia fizyczne.
- Każdy rozdział jest starannie przygotowywany. Zjawiska fizyczne opisane są za pomocą definicji, wzorów oraz jeśli istnieje potrzeba, ilustracji.
- Dostępne są dwie wersje językowe aplikacji: w języku polskim oraz angielskim.

Wadą tej aplikacji, podobnie jak w przypadku aplikacji Pierwiastki jest monotematyczność, oraz brak elementu gry.



Rysunek 1.13. Przykładowe rozdziały zagadnień fizyki.



Rysunek 1.14 Przykład działu aplikacji Fizyka na 5 – dział Rzut ukośny.



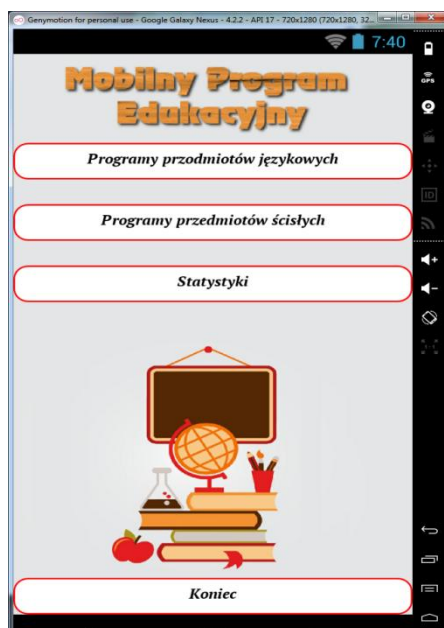
Rysunek 1.15 Przykład działu aplikacji Fizyka na 5 – dział Rzut optyka falowa.

Rozdział 2. Szczegółowy opis aplikacji Mobilny Program Edukacyjny

W niniejszym rozdziale przedstawiono wszystkie moduły jakie zostały zaimplementowane w aplikacji stworzonej na potrzeby pracy. Moduł pierwszy to główny ekran aplikacji. To z jego poziomu można wybrać pozostałe moduły programu: *Programy przedmiotów językowych*, *Programy przedmiotów ścisłych* oraz *Statystyki*. Ostatnim modulem jest moduł wyjścia z aplikacji. Moduł pierwszy i drugi dzielą się na pod moduły edukacyjne. W zależności od tematyki, każdy z nich został przydzielony do pierwszego lub drugiego modułu. *Moduł Przedmiotów językowych* zawiera pod moduły: *Ortografia dla najlepszych*, *Wisielec z Polski*, *Wisielec z Anglii*, *Nauka angielskiego*. *Moduł Przedmiotów ścisłych* zawiera pod moduły: *Szybkie operacje*, *Tablica Mendelejewa*, *Tablica Mendelejewa ENG* oraz pod moduł *Systemy liczbowe*.

2.1 Moduł 1: Główny ekran aplikacji

Moduł przedstawia wszystkie funkcje aplikacji Mobilny Program Edukacyjny. Z poziomu tego modułu użytkownik wybiera jeden z czterech interesujących go pod modułów. Dostęp do tych modułów znajduje się pod przyciskami. Klikając na jeden z nich użytkownik przenosi się do wybranego modułu. Rysunek 2.1 przedstawia główne okno aplikacji. Użytkownik po wybraniu dowolnego modułu (za wyjątkiem ostatniego), może w każdym momencie powrócić do ekranu głównego. Odbywa się to za pomocą przycisku telefonu wstecz.



Rysunek 2.1 Główne okno aplikacji.

2.2 Moduł 2: Programy przedmiotów językowych

W tym podrozdziale zostaną omówione wszystkie moduły dotyczące przedmiotów językowych. Po wybraniu modułu językowego, użytkownik będzie miał wyświetloną na ekranie listę dostępnych modułów językowych. Każdy moduł reprezentowany jest na liście pod postacią ikony, nazwy i krótkiego opisu. Rysunek 2.2 przedstawia listę dostępnych pod modułów językowych.



Rysunek 2.2 Lista dostępnych pod modułów językowych.

2.2.1 Moduł *Ortografia dla najlepszych*

Moduł *Ortografia dla najlepszych*, ma na celu rozwijać u użytkownika zdolności ortograficzne z Języka polskiego. Moduł ten jest grą. Gracz ma za zadanie stwierdzić, czy podany wyraz pisany jest np. przez 'ó' czy przez 'u'. Wyrazy są niekiedy tak dobrane, że ich częstość występowania w języku polskim jest bardzo rzadka. Dlatego nawet użytkownicy znający biegle zasady polskiej ortografii mogą mieć problem z udzieleniem właściwej odpowiedzi.

W momencie uruchomienia modułu *Ortografia dla najlepszych* zostaje wylosowane graczowi zadanie. Na środku ekranu urządzenia zostaje wyświetlony napis. Napis ten jest polskim wyrazem, w którym w miejsce wystąpienia znaku ortograficznego wstawiono trzy kropki. Na Rysunku 2.3 oznaczono czerwoną elipsą wyraz z którego ukryto znak ortograficzny.



Rysunek 2.3 Przedstawia przykład wylosowanego wyrazu do odgadnięcia.

Wzorując się na przykładzie z Rysunku 2.3, użytkownik ma dwie możliwości udzielenia odpowiedzi. Wskazać na przycisk oznaczony znakiem ortograficznym 'rz', albo na przycisk oznaczony znakiem ortograficznym 'ż'. Użytkownik po wybraniu odpowiedzi, nie może już jej zmienić. W momencie udzielenia odpowiedzi użytkownik otrzymuje komunikat w postaci animacji, czy odpowiedział poprawnie, czy też wybrana odpowiedź jest niepoprawna. W przypadku udzielenia poprawnej odpowiedzi, wylosowany wyraz zmienia swoją barwę z koloru czarnego na kolor zielony. Równocześnie pojawia się charakterystyczna animacja zwijania tekstu w pionie. Jeżeli użytkownik wybrał odpowiedź niepoprawną, wylosowany wyraz zmienia swoją barwę z koloru czarnego na kolor czerwony. Również i w tym przypadku pojawia się charakterystyczna animacja wstrząsania tekstu. Rysunek 2.4 oraz Rysunek 2.5 przedstawiają dwa różne przypadki udzielenia odpowiedzi. Na rysunkach udało się wyłącznie pokazać zamianę koloru.



Rysunek 2.4 Przypadek udzielenia poprawnej odpowiedzi.



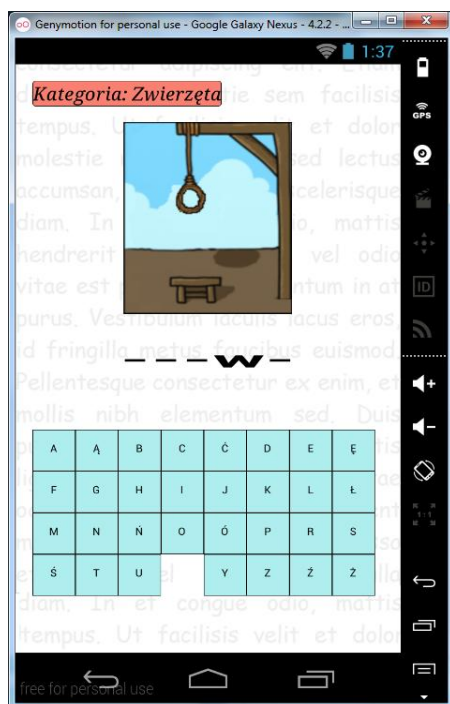
Rysunek 2.5 Przypadek udzielenia niepoprawnej odpowiedzi.

2.2.2 Moduł Wisielec z Polski

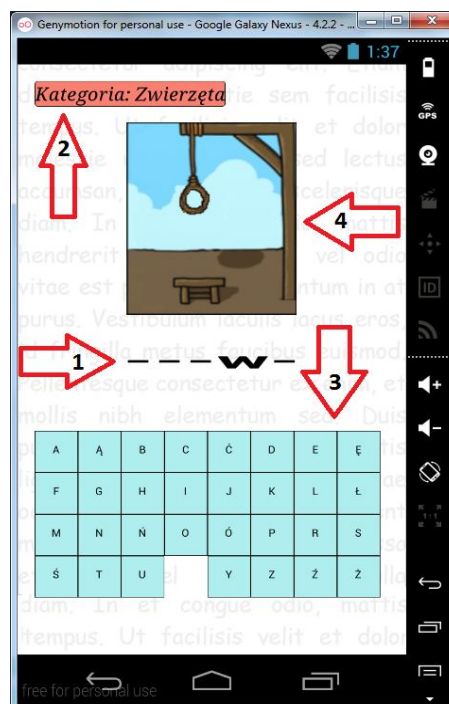
Moduł *Wisielec z Polski* jest grą edukacyjną mającą na celu poszerzanie wiedzy użytkownika z zakresu rzadkich wyrazów Języka polskiego. Gra jest wzorowana na klasycznej grze papierowej Wisielec (znana również pod nazwami Szubienica i Powieszony).

W momencie uruchomienia modułu, automatycznie zostaje uruchomiona nowa rozgrywka. Przedstawia to Rysunek 2.6. Wylosowany zostaje pojedynczy wyraz, który będzie do odgadnięcia w trakcie gry. Wyraz ten należy do z góry ustalonej kategorii tematycznej (np. kategoria planety, kategoria owoce). Rysunek 2.7 (strzałka nr 2) przedstawia kategorię przykładowo wylosowanego wyrazu. Graczowi na starcie nowej rozgrywki zostaje podana tylko jedna, losowo wybrana litera należąca do wyrazu i jest ona odkrywana dla wszystkich jej wystąpień w wyrazie. Reszta liter jest ukryta pod znakiem myślnika. Rysunek 2.7 (strzałka nr 1) przedstawia wylosowany wyraz wraz z odsłoniętym znakiem. Rysunek 2.7 (strzałka nr 3) przedstawia przyciski oznaczone literkami, których gracz będzie używał do odgadywania znaków wylosowanego wyrazu. W momencie wybrania znaku, przycisk oznaczony wybranym znakiem zostaje wyłączony z dalszej rozgrywki. Jeżeli wybrany znak należy do wylosowanego wyrazu, następuje jego odsłonięcie we wszystkich jego wystąpieniach w danym wyrazie, a gracz zachowuje komplet szans. Jeżeli wybrany znak nie należy do wylosowanego wyrazu,

gracz traci jedną szansę. Ponadto zmienia się obraz wisielca w zależności od tego ile pozostało graczowi szans. Rysunek 2.7 (strzałka nr 4) przedstawia aktualny obraz wisielca, w zależności od ilości pozostałych szans. Tabela 2.1 Przedstawia etapy zmiany obrazu wisielca, przy podawaniu błędnych znaków. Gracz od momentu startu ma pięć szans. Rozgrywka kończy się wygraną jeżeli gracz ma jedną lub więcej szans i wszystkie litery zostały odsłonięte. Gra kończy się przegraną jeżeli graczowi pozostało zero szans.






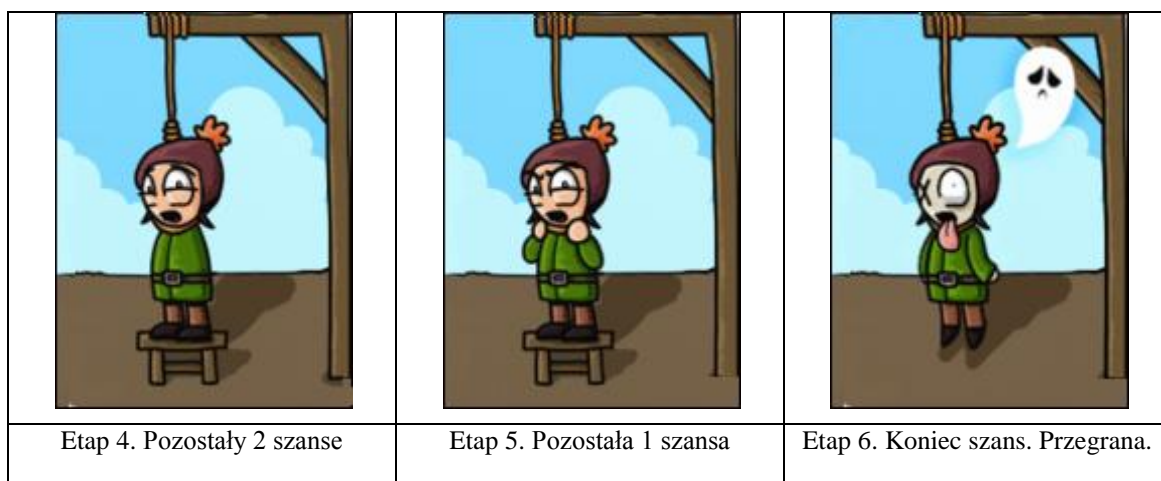
Rysunek 2.6 Wygląd modułu *Wiselec z Polski* przy rozpoczęciu nowej gry.



Rysunek 2.7 Wygląd modułu *Wiselec z Polski* przy rozpoczęciu nowej gry wraz z oznaczeniami.

Tabela 2.1 Etapy zmiany obrazu wisielca, przy podawaniu błędnych znaków.

		
Etap 1. Pozostało 5 szans. Początek nowej gry	Etap 2. Pozostały 4 szanse	Etap 3. Pozostały 3 szanse



Rysunek 2.8 Ekran wygranej.



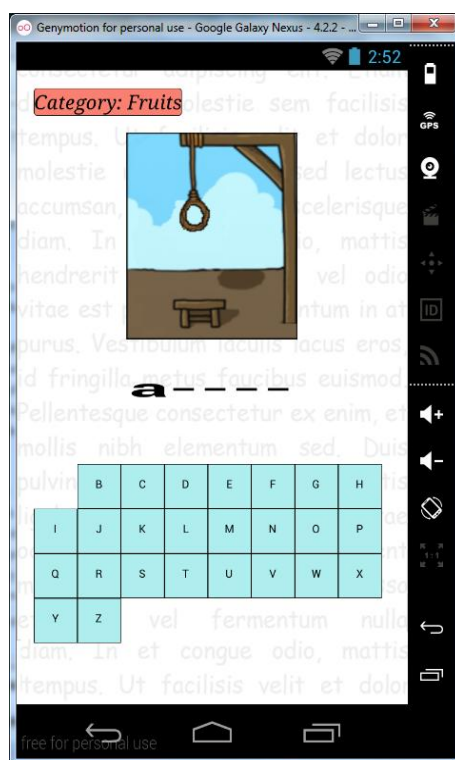
Rysunek 2.9 Ekran przegranej.

Bez względu na to, czy użytkownik wygrał daną rozgrywkę (Rysunek 2.8), czy przegrał (Rysunek 2.9), okno aplikacji zostaje po kilku sekundach automatycznie odświeżone, a gracz może rozpocząć nową rozgrywkę.

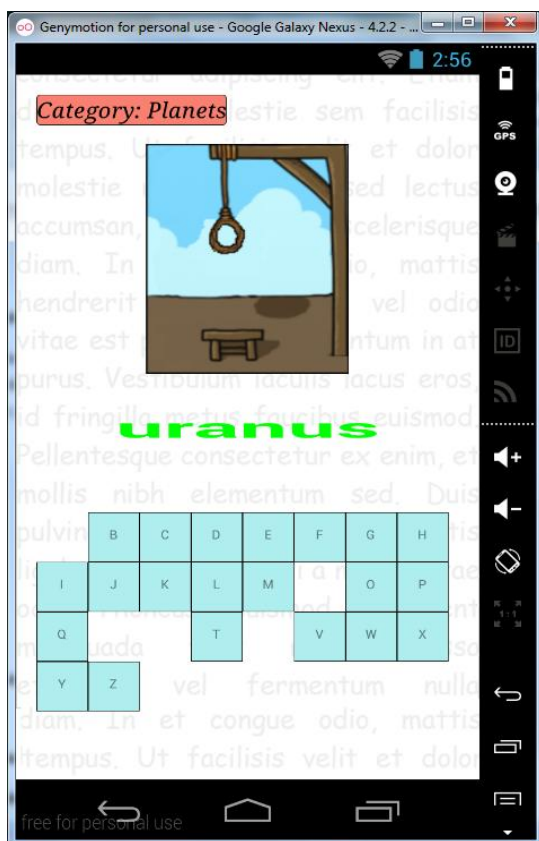
2.2.3 Moduł Wisielec z Anglii

Moduł *Wisielec z Anglii* jest grą edukacyjną mającą na celu poszerzanie wiedzy użytkownika z zakresu rzadkich wyrazów języka angielskiego. Gra jest wzorowana na klasycznej grze papierowej Wisielec (znana również pod nazwami Szubienica i Powieszony).

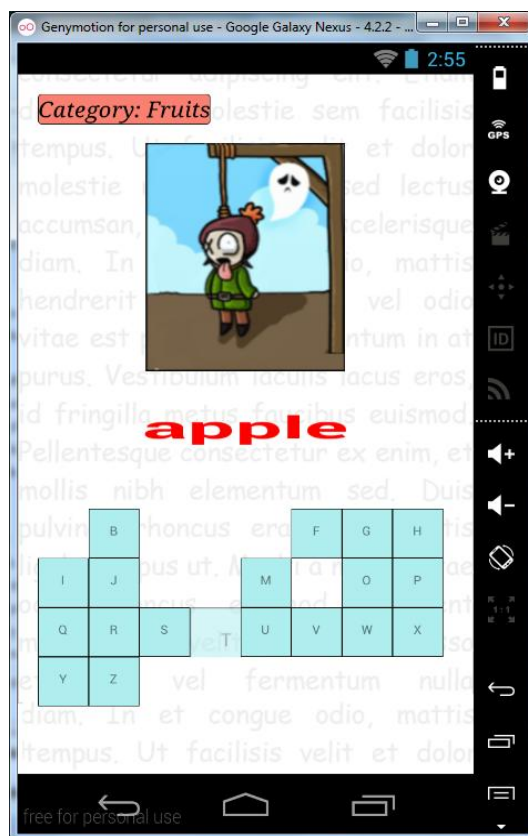
Rozgrywka przebiega w identyczny sposób jak w przypadku modułu *Wisielca z Polski*. Różnicą jest tylko język prowadzonej rozgrywki. W tym module zarówno losowane słowa jak i kategorie są w języku angielskim. Dostępne znaki pod przyciskami również zostały dostosowane do języka angielskiego. Rysunek 2.10 przedstawia ekran nowej rozgrywki dla *Wisielca z Anglii*. Rysunek 2.11 przedstawia ekran wygranej rozgrywki. Rysunek 2.12 przedstawia ekran przegranej rozgrywki.



Rysunek 2.10 Wygląd modułu *Wisielec z Anglii* przy rozpoczęciu nowej rozgrywki.



Rysunek 2.11 Ekran wygranej.

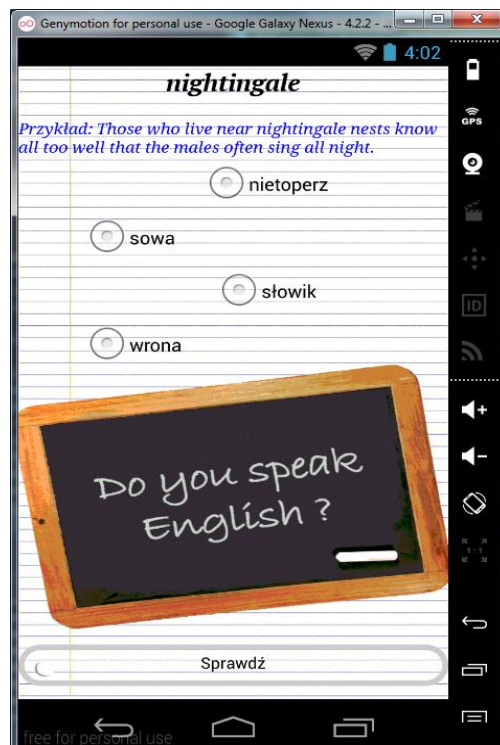


Rysunek 2.12 Ekran przegranej.

2.2.4 Moduł Nauka angielskiego

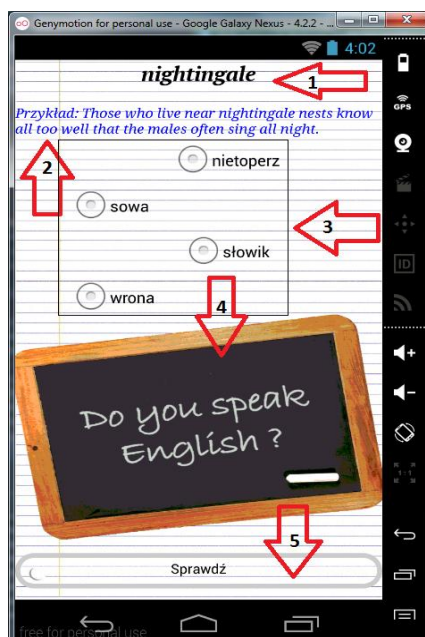
W module *Nauka angielskiego*, użytkownik uczy się trudniejszych wyrazów języka angielskiego poprzez dopasowanie do wylosowanego polskiego słowa jego angielskiego odpowiednika. Wyrazy te mogą być rzadko używane w codziennym życiu, dlatego do każdego wylosowanego słowa, wyświetlany jest jego praktyczny przykład użycia w zdaniu. Dzięki temu użytkownik niekiedy może wydedukować z kontekstu znaczenie wylosowanego słowa i odpowiedzieć prawidłowo na pytanie.

W momencie uruchomienia modułu *Nauka angielskiego* zostaje automatycznie uruchomiona rozgrywka. Rysunek 2.13 przedstawia ekran nowej rozgrywki dla tego modułu.



Rysunek 2.13 Ekran nowej rozgrywki *Nauka angielskiego*.

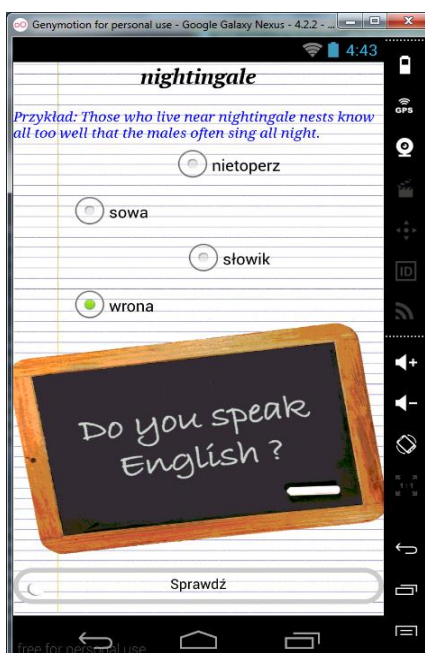
Po rozpoczęciu gry na ekranie pojawia się nowo wylosowany wyraz wraz z jego przykładowym użyciem w zdaniu. Rysunek 2.14 (strzałka nr 1) przedstawia miejsce wyświetlenia wylosowanego wyrazu. Strzałka nr 2 przedstawia miejsce wyświetlenia przykładowego zdania. Zadaniem użytkownika jest wybranie jednej z czterech proponowanych odpowiedzi. Kolejność ich występowania jest losowa. Na rysunku 2.14 (strzałka nr 3) przedstawiono w czarnej, prostokątnej otoczce miejsce wyświetlenia proponowanych odpowiedzi. Po wybraniu jednej z czterech proponowanych odpowiedzi, użytkownik naciska przycisk *Sprawdź* znajdujący się w dolnej części ekranu. Strzałka nr 5 przedstawia miejsce przycisku zatwierdzania wybranej odpowiedzi. W zależności od poprawności wybranej odpowiedzi, na tablicy (strzałka nr 4) pojawia się stosowny komunikat o tym, czy użytkownik odpowiedział poprawnie, czy udzielił niepoprawnej odpowiedzi.



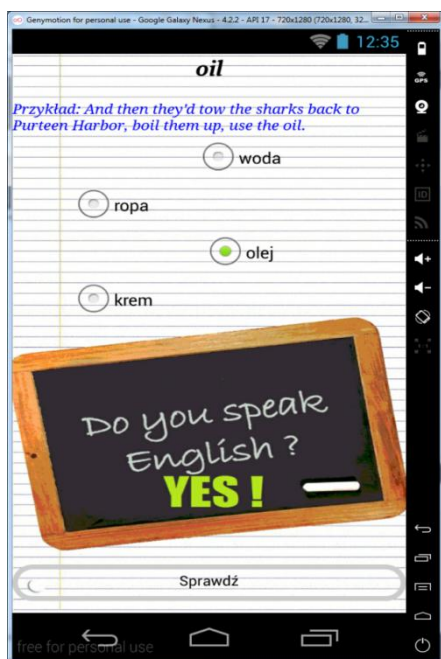
Rysunek 2.14 Ekran nowej rozgrywki *Nauka angielskiego* wraz z oznaczeniami.

Użytkownik przed zatwierdzeniem może zmienić wybraną odpowiedź na inną. Rysunek 2.15 przedstawia wybraną odpowiedź przez użytkownika przed jej zatwierdzeniem.

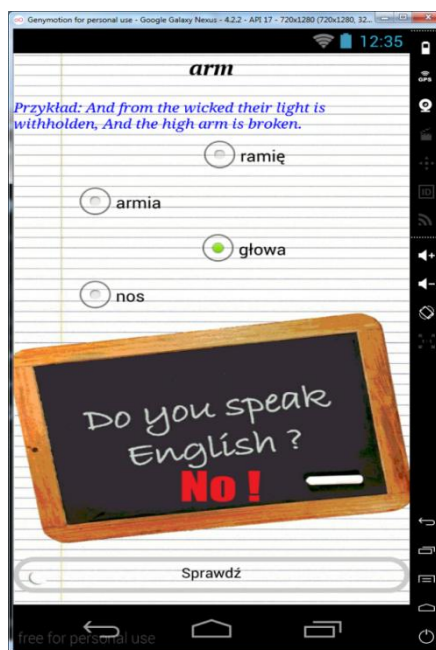
Na Rysunku 2.16 przedstawiono ekran, na którym pojawił się komunikat o poprawnie udzielonej odpowiedzi. Na Rysunku 2.17 przedstawiono ekran, na którym pojawił się komunikat o niepoprawnie udzielonej odpowiedzi.



Rysunek 2.15 Ekran rozgrywki przed zatwierdzeniem wybranej odpowiedzi.



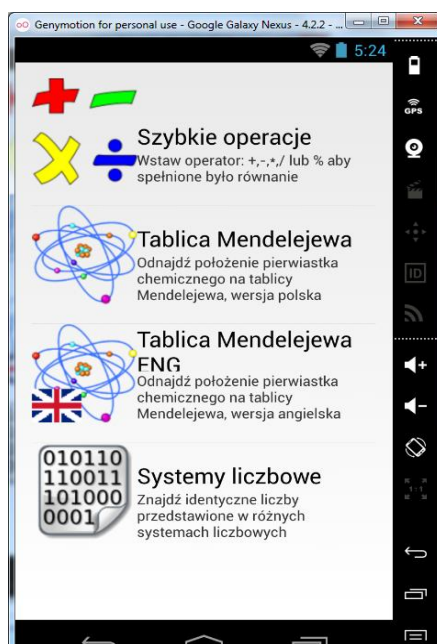
Rysunek 2.16 Ekran poprawnej odpowiedzi.



Rysunek 2.17 Ekran niepoprawnej odpowiedzi.

2.3 Moduł 3: Programy przedmiotów ścisłych

W tym podrozdziale zostaną omówione wszystkie moduły dotyczące przedmiotów ścisłych. Po wybraniu modułu przedmiotów ścisłych, użytkownik będzie miał wyświetloną na ekranie listę dostępnych modułów przedmiotów. Każdy moduł reprezentowany jest na liście pod postacią ikony, nazwy i krótkiego opisu. Rysunek 2.18 przedstawia listę dostępnych pod modułów przedmiotów ścisłych.



Rysunek 2.18 Lista dostępnych modułów przedmiotów ścisłych.

2.3.1 Moduł Szybkie operacje

Moduł *Szybkie operacje* jest grą edukacyjną, dzięki której użytkownik biegle wykształca swoje zdolności arytmetyczne.

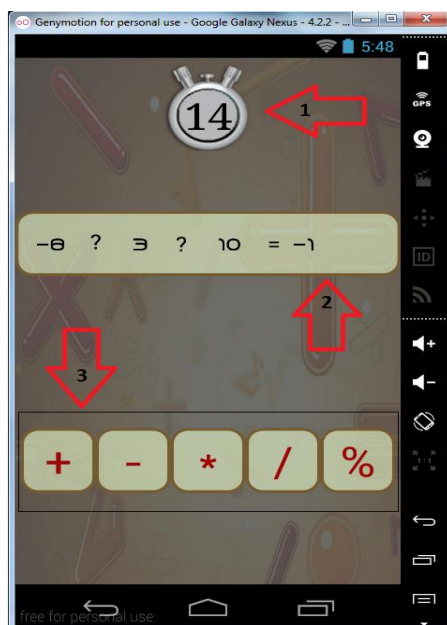
W momencie uruchomienia modułu *Szybkie operacje* zostaje automatycznie uruchomiona nowa rozgrywka. Rysunek 2.19 przedstawia ekran nowo rozpoczętej rozgrywki.



Rysunek 2.19 Ekran nowej rozgrywki modułu *Szybkie operacje*.

Na ekranie użytkownika generowane zostaje równanie matematyczne, w którym w miejsce operatorów wstawiono znak '?'. Na Rysunku 2.20 (strzałka nr 2) pokazano miejsce, w którym wyświetlane jest równanie. Gracz za pomocą przycisków oznaczonych symbolami: '+', '-', '*', '/' oraz '%' wstawia wybrany operator w miejsce animowanego pytajnika. Strzałka nr 3 pokazuje miejsce, w którym znajdują się przyciski. Tabela 2.2 przedstawia znaczenie operatorów, dostępnych w module. Charakterystyczna animacja znaku '?' informuje użytkownika, w którym miejscu równania zostanie wstawiony wybrany operator. Po wybraniu obu operatorów, następuje weryfikacja równania, zgodnie z priorytetami operatorów arytmetycznych. Możliwe jest zaistnienie takiej sytuacji, w której istnieje więcej niż jedna para operatorów spełniająca równanie. Wówczas jest to uwzględniane jako poprawne rozwiązanie zadania. Każda rozgrywka jest ograniczona, czasem piętnastu sekund.

Jeżeli gracz nie udzieli odpowiedzi w zadany czas, wówczas gracz ponosi porażkę. Strzałka nr 1 na Rysunku 2.20 ukazuje zegar odliczający pozostały czas.



Rysunek 2.20 Przedstawia ekran rozgrywki, wraz z oznaczeniami.

Tabela 2.2 Znaczenie operatorów arytmetycznych dostępnych w module *Szybkie operacje*.

+	Dodawanie
-	Odejmowanie
*	Mnożenie
/	Dzielenie całkowite
%	Reszta z dzielenia całkowitego

Rozgrywka kończy się w przypadku, gdy upłynie czas na udzielenie odpowiedzi, bądź użytkownik zdąży wstawić oba operatory w zadany czas. Jeżeli równanie jest poprawne, wówczas otoczka równania zmienia swoją barwę na kolor zielony. Rysunek 2.21 przedstawia ekran poprawnie udzielonej odpowiedzi. W przeciwnym wypadku, otoczka zmienia swoją barwę na kolor czerwony. Rysunek 2.22 przedstawia ekran niepoprawnej udzielonej odpowiedzi. W obu przypadkach, po zakończeniu rozgrywki, rozpoczyna się nowa gra i zostaje wygenerowane nowe równanie. Zegar ponownie rozpoczyna odliczanie od piętnastu sekund.



Rysunek 2.21 Ekran poprawnej odpowiedzi.

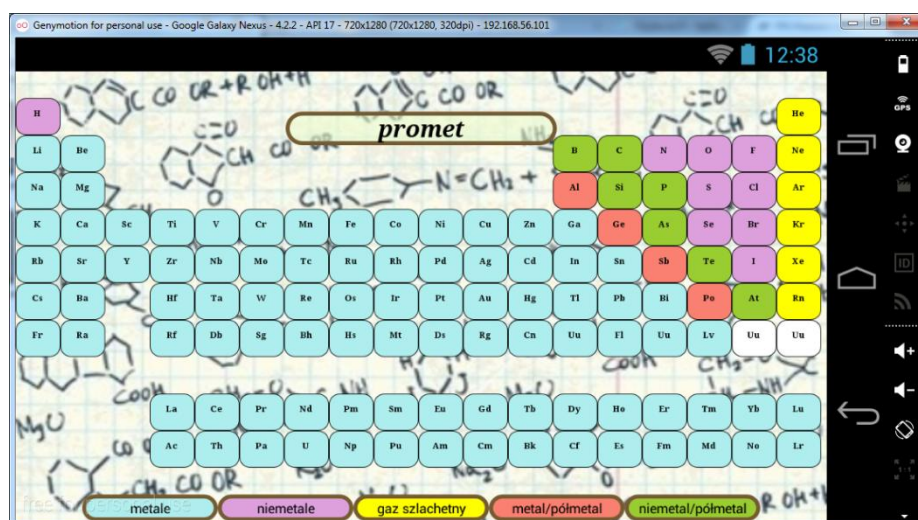


Rysunek 2.22 Ekran niepoprawnej odpowiedzi.

2.3.2 Moduł Tablica Mendelejewa

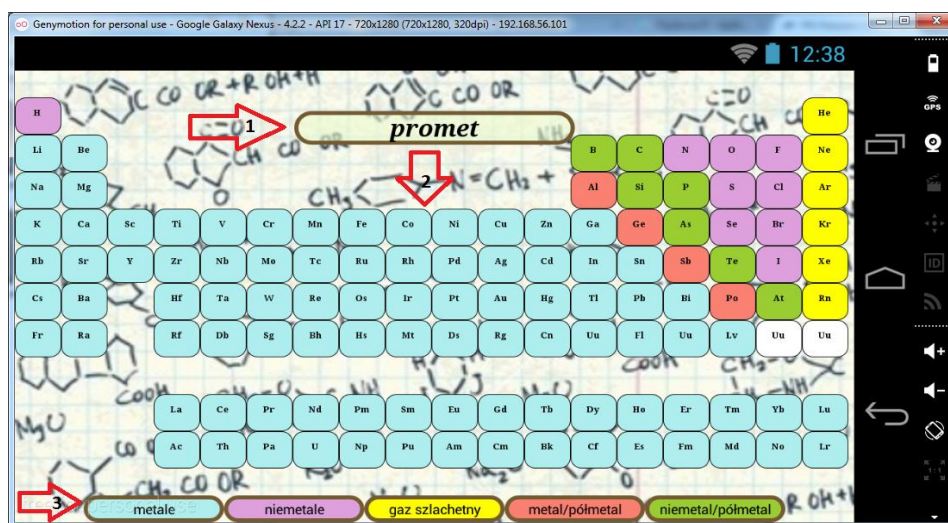
Moduł *Tablica Mendelejewa* jest grą edukacyjną, mającą na celu utrwalać wiedzę gracza w zakresie rozpoznawania pierwiastków chemicznych. Umożliwia łatwe kojarzenie nazw pierwiastka z jego symbolem oraz położeniem w układzie okresowym pierwiastków.

W momencie uruchomienia modułu *Tablica Mendelejewa* zostaje automatycznie uruchomiona nowa rozgrywka. Rysunek 2.23 przedstawia ekran nowo rozpoczętej rozgrywki.



Rysunek 2.23 Ekran nowej rozgrywki modułu *Tablica Mendelejewa*.

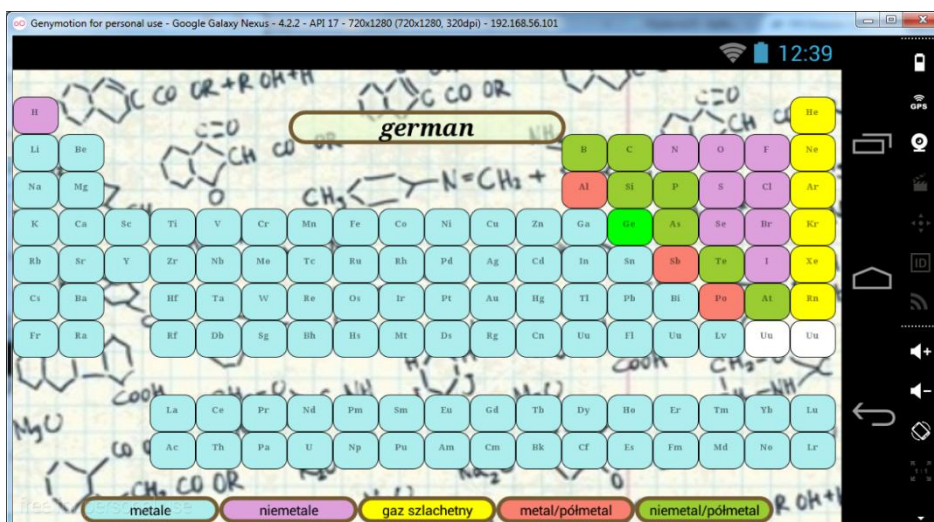
Na ekranie urządzenia zostaje wylosowana nazwa pierwiastka chemicznego. Rysunek 2.24 (strzałka nr 1) przedstawia miejsce wyświetlenia nazwy pierwiastka. Zadaniem użytkownika jest dopasowanie nazw pierwiastków do ich symboli znajdujących się pod przyciskami ułożonymi na wzór tablicy Mendelejewa. Strzałka nr 2 przedstawia symbole pierwiastków chemicznych. Dla urozmaicenia szaty graficznej, każdy symbol znajdujący się na tablicy Mendelejewa został pokolorowany w zależności od charakteru chemicznego pierwiastka. Strzałka nr 3 wskazuje na kolory pierwiastków w tablicy Mendelejewa w zależności od ich charakteru chemicznego.



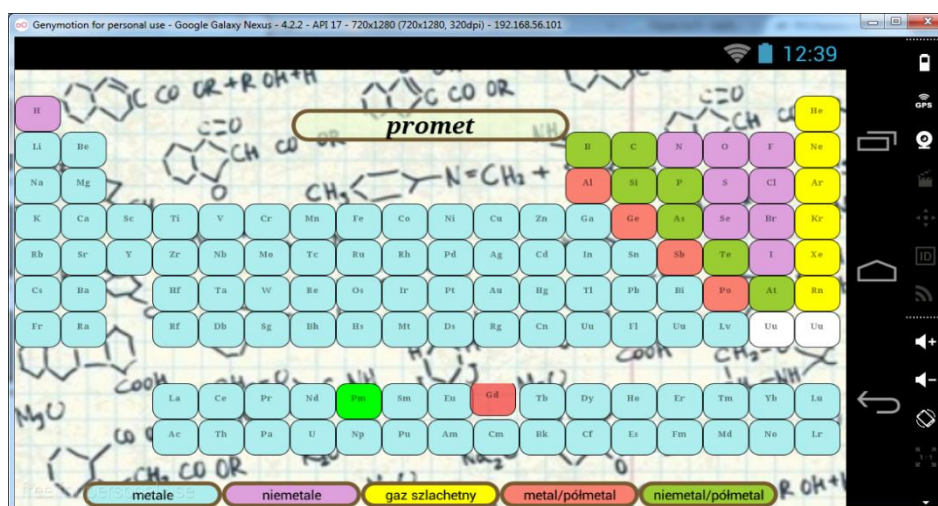
Rysunek 2.24 Ekran nowej rozgrywki modułu Tablica Mendelejewa, wraz z oznaczeniami.

Po wybraniu symbolu pierwiastka, następuje weryfikacja czy wybrany symbol jest symbolem wylosowanego pierwiastka. Jeżeli symbol jest symbolem wylosowanego pierwiastka, wówczas gracz poprawnie rozwiązał zadanie. Towarzyszy temu charakterystyczna animacja wybranego symbolu: przycisk pod jakim się znajduje, zmienia naprzemiennie swój kolor ze swojego pierwotnego koloru na kolor zielony. Rysunek 2.25 przedstawia ekran poprawnie udzielonej odpowiedzi. Następnie następuje weryfikacja, czy wybrany symbol jest symbolem wylosowanego pierwiastka. Jeżeli symbol nie jest symbolem wylosowanego pierwiastka, wówczas gracz niepoprawnie rozwiązał zadanie. Towarzyszy temu charakterystyczna animacja wybranego symbolu: przycisk pod jakim się znajduje, zmienia naprzemiennie swój kolor ze swojego pierwotnego koloru na kolor czerwony. Natomiast pierwiastek pod którym znajdowała się poprawna odpowiedź, zmienia naprzemiennie swój kolor ze swojego pierwotnego koloru na kolor zielony. Rysunek 2.26 przedstawia ekran niepoprawnie udzielonej

odpowiedzi. Po zakończeniu zadania, niezależnie od jego wyniku, ekran jest odświeżany i rozpoczyna się nowa rozgrywka.



Rysunek 2.25 Ekran poprawnej odpowiedzi.

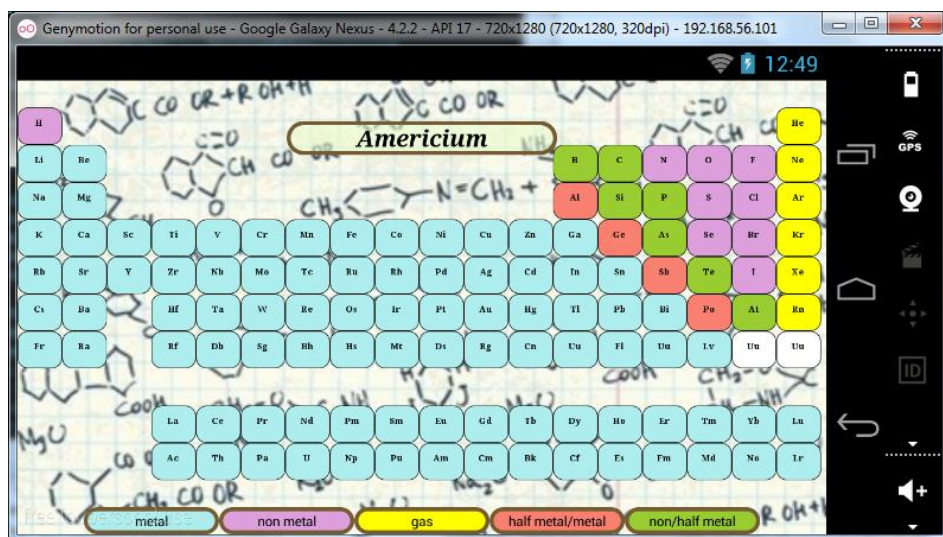


Rysunek 2.26 Ekran niepoprawnej odpowiedzi.

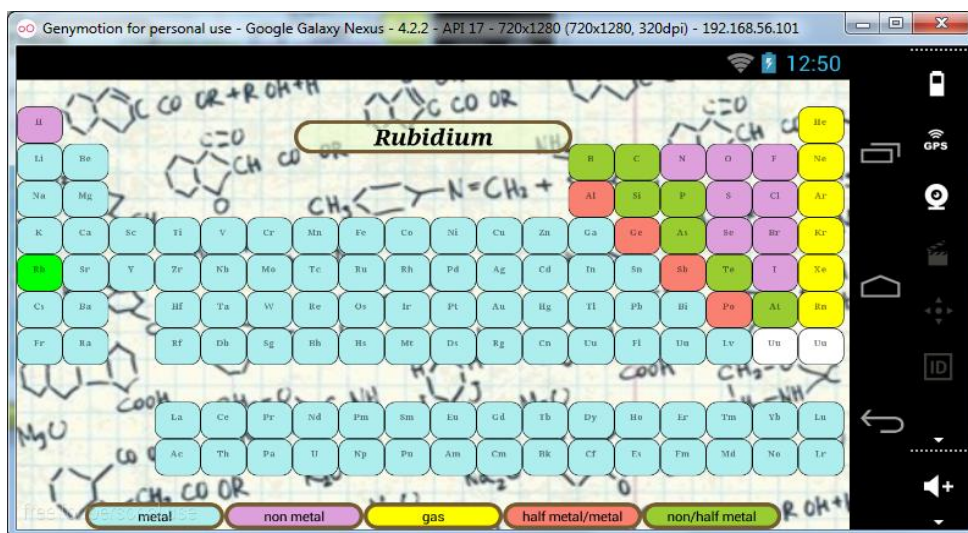
2.3.3 Moduł Tablica Mendelejewa ENG

Moduł *Tablica Mendelejewa ENG* jest grą edukacyjną w języku angielskim, mającą na celu utrwalac wiedzę gracza w zakresie rozpoznawania pierwiastków chemicznych.

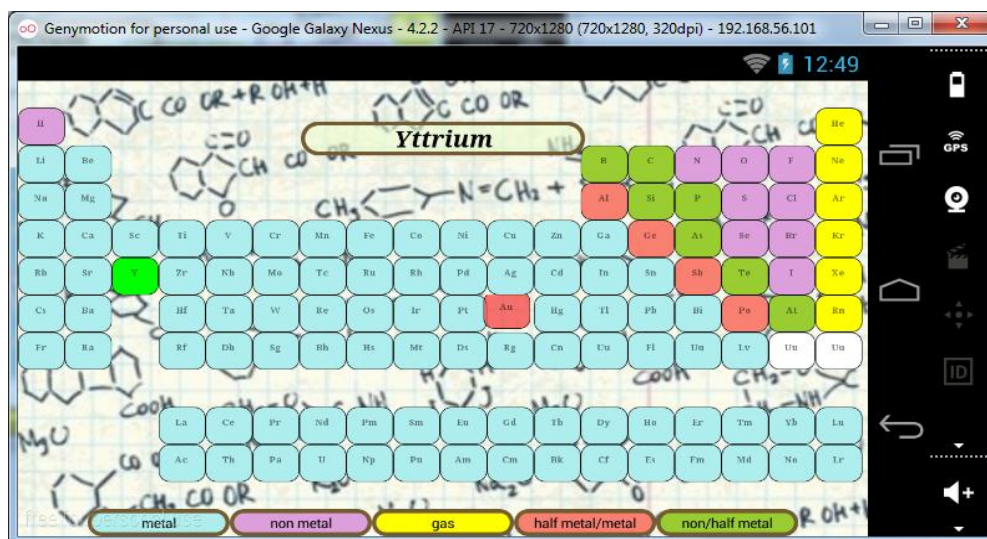
Rozgrywka przebiega w identyczny sposób jak w przypadku modułu *Tablica Mendelejewa*. Różnicą jest tylko język prowadzonej rozgrywki. W tym module zarówno losowane słowa jak i ich legenda są w języku angielskim. Rysunek 2.27 przedstawia ekran nowej rozgrywki. Rysunek 2.28 przedstawia ekran poprawnej odpowiedzi. Rysunek 2.29 przedstawia ekran niepoprawnej odpowiedzi.



Rysunek 2.27 Ekran nowej rozgrywki modułu *Tablica Mendelejewa ENG*.



Rysunek 2.28 Ekran poprawnej odpowiedzi.

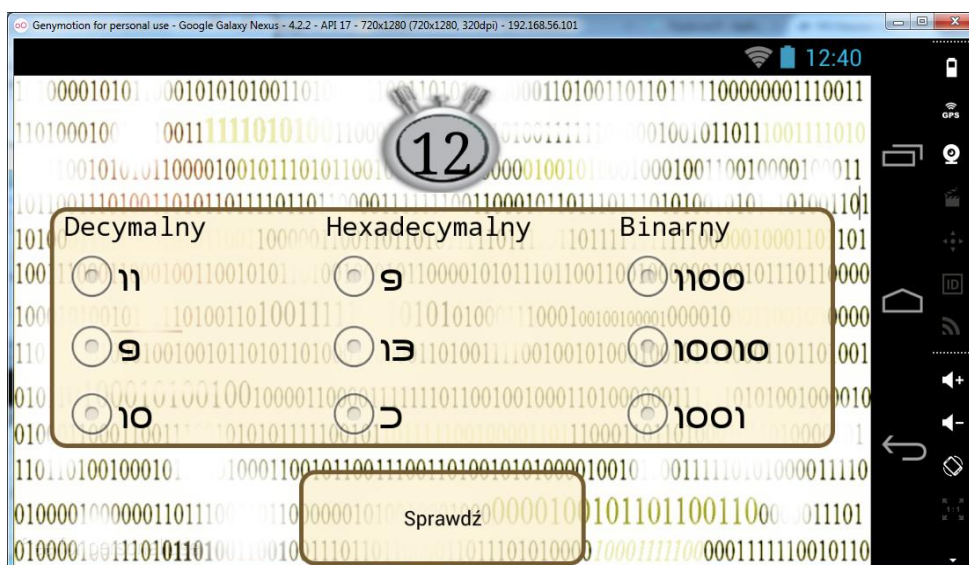


Rysunek 2.29 Ekran niepoprawnej odpowiedzi.

2.3.4 Moduł Systemy liczbowe

Moduł *Systemy liczbowe* jest grą edukacyjną, dzięki której gracz uczy się konwersji liczb w trzech różnych systemach liczbowych: system dziesiętny (decymalny), system szesnastkowy (hexadecymalny) oraz system dwójkowy (binarny) .

W momencie uruchomienia modułu *Systemy liczbowe* na ekranie urządzenia zostaje automatycznie uruchomiona nowa rozgrywka. Rysunek 2.37 przedstawia ekran nowo rozpoczętej rozgrywki.

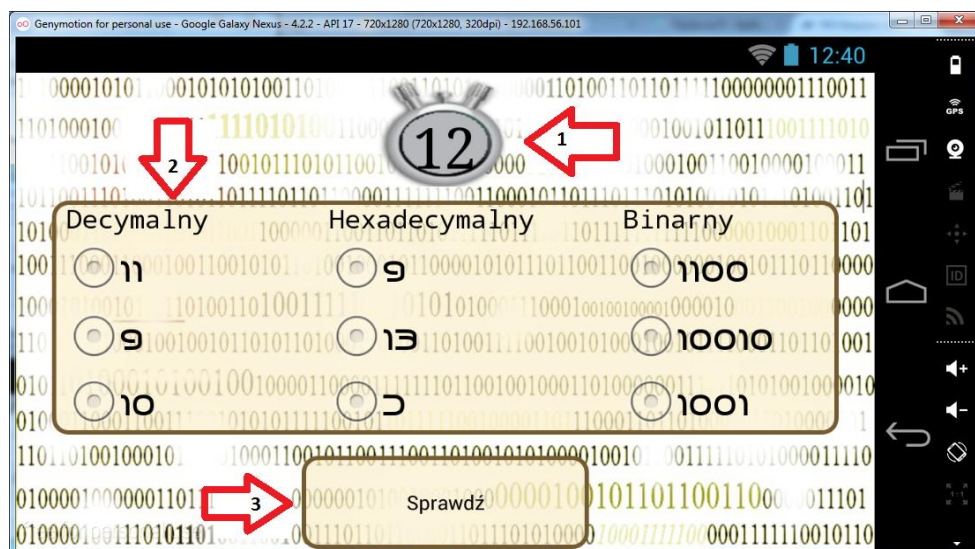


Rysunek 2.30 Ekran nowej rozgrywki modułu *Szybkie operacje*.

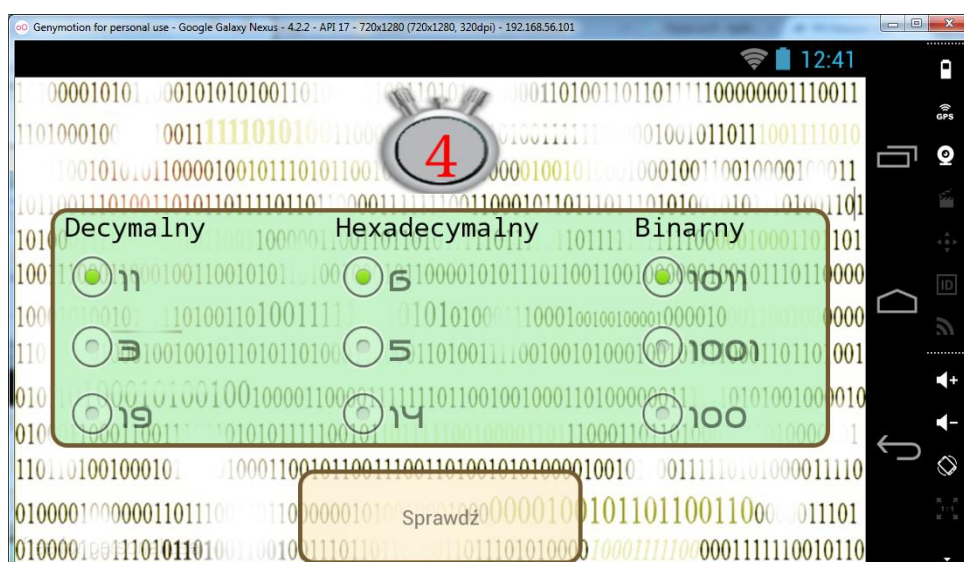
Na ekranie urządzenia zostają wylosowane dla każdego systemu liczbowego trzy różne liczby. Rysunek 2.31 (strzałka nr 2) przedstawia wylosowane liczby w odpowiednich dla nich systemach liczbowych.

Zadaniem gracza jest znalezienie trzech równych sobie liczb przedstawionych w różnych systemach liczbowych. Istnieje tylko jedna poprawna konfiguracja liczb. Gracz zaznacza z każdego systemu liczbowego jedną odpowiedź, po czym zatwierdza ją przyciskiem *Sprawdź*. Strzałka nr 3 przedstawia przycisk sprawdzający zadanie. Dodatkowo każde zadanie ograniczone jest czasem piętnastu sekund. Przed zatwierdzeniem odpowiedzi, gracz może dowolnie zmieniać wybrane odpowiedzi, o ile nie skończył mu się czas. Strzałka nr 1 przedstawia zegar aktualnej rozgrywki. Rozgrywka kończy się w przypadku, gdy upłynie czas na udzielenie odpowiedzi, bądź użytkownik zdąży rozwiązać zadanie w zadanym czasie. Jeżeli odpowiedź jest poprawna, wówczas otoczka z liczbami zmienia swoją barwę na kolor zielony. Rysunek

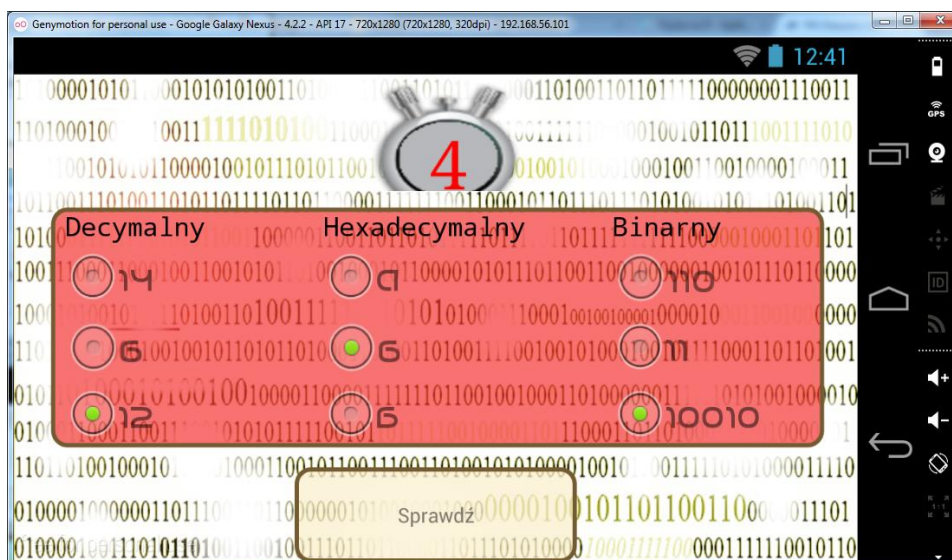
2.32 przedstawia ekran poprawnie udzielonej odpowiedzi. W przeciwnym wypadku, otoczka zmienia swoją barwę na kolor czerwony. Rysunek 2.33 przedstawia ekran niepoprawnie udzielonej odpowiedzi. W obu przypadkach, po zakończeniu rozgrywki, rozpoczyna się nowa gra. Zostają wygenerowane nowe liczby. Zegar ponownie rozpoczyna odliczanie od piętnastu sekund.



Rysunek 2.31 Ekran nowej rozgrywki modułu *Szybkie operacje*, wraz z oznaczeniami.



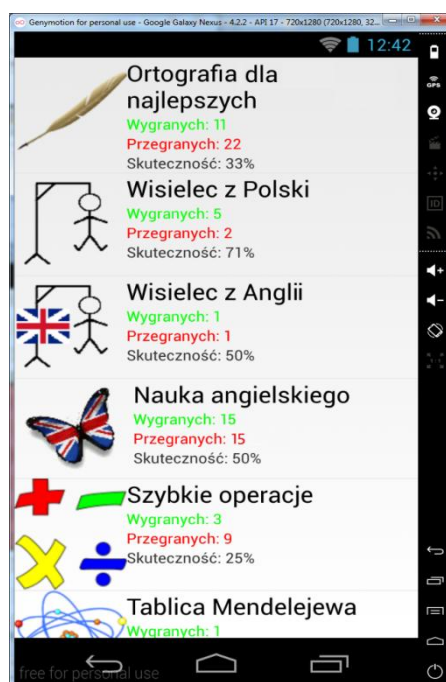
Rysunek 2.32 Ekran poprawnej odpowiedzi.



Rysunek 2.33 Ekran niepoprawnej odpowiedzi.

2.4 Moduł 4: Statystyki

Moduł *Statystyki* służy do prowadzenia statystyk dla każdej gry edukacyjnej. Po każdej zakończonej grze, statystyki są aktualizowane. Do pojedynczej statystyki zalicza się: nazwę gry edukacyjnej, jej logo, ilość wygranych rozgrywek, ilość przegranych rozgrywek oraz stosunek wygranych do ilości rozegranych gier. Rysunek 2.34 przedstawia ekran modułu *Statystyki*.



Rysunek 2.34 Ekran modułu *Statystyki*.

2.5 Moduł 5: Koniec

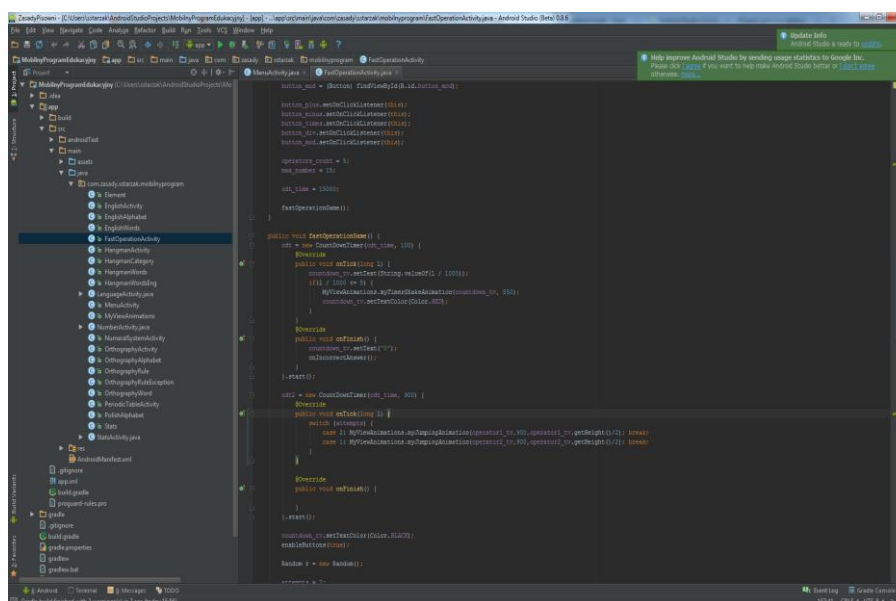
Moduł *Koniec* nie zawiera żadnych pod modułów i służy jedynie do zakończenia działania aplikacji.

Rozdział 3: Wykorzystane technologie programistyczne

W tym rozdziale opisane zostaną technologie programistyczne jakie zostały użyte przy tworzeniu programu Mobilny Program Edukacyjny. Wprowadzone zostaną podstawowe pojęcia związane z programowaniem na platformę Android. Zilustrowane będą tabele baz danych, oraz przedstawione algorytmy modułów w postaci opisu oraz kodu źródłowego.

3.1 Środowisko programistyczne

Do stworzenia aplikacji wykorzystane zostało środowisko programistyczne Android Studio for Windows [11], które przeznaczone jest dla developerów aplikacji na platformę Android. Środowisko to pozwala na wygodne projektowanie i tworzenie własnych aplikacji dedykowanych systemowi Android. Rysunek 3.1 przedstawia okno środowiska programistycznego Android Studio.



Rysunek 3.1 Środowisko programistyczne Android Studio.

3.2 Języki programowania

3.2.1 Język Java

Aplikacja Mobilny Program Edukacyjny stworzona została w języku Java [6]. Głównymi cechami języka Java są [7]:

- zorientowanie obiektowe,
- jednokrotne dziedziczenie,
- przenośność, niezależność od architektury i systemu operacyjnego,

- wsparcie dla programowania współbieżnego, wielowątkowość,
- obsługa błędów poprzez wyjątki wymuszana przez kompilator,
- liczne biblioteki (pakiety) standardowe i zewnętrzne,
- kompilowanie do postaci kodu pośredniego,
- zarządzanie pamięcią, odzyskiwanie nieużytków,
- brak jawnych operacji na wskaźnikach pamięci,
- silna kontrola typów,
- wielopoziomowe mechanizmy bezpieczeństwa.

3.2.2 Język XML

W projektowaniu elementów graficznych (układów, przycisków, widoku tekstów, widoku obrazów) interfejsu użytkownika aplikacji Mobilny Program Edukacyjny wykorzystany został język XML [8]. Język XML jest uniwersalnym językiem znaczników przeznaczonym do reprezentowania różnych danych w strukturalizowany sposób. Na Listingu 3.1 przedstawiono przykładowy kod XML użyty w module *Statystyki* do utworzenia widoku dla tego modułu.

```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent">

<ImageView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:id="@+id/imageView"
android:src="@drawable/ic_launcher"
android:layout_alignParentTop="true" />

<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:textAppearance="?android:attr/textAppearanceLarge"
android:text="Large Text"
android:id="@+id/textView"
android:layout_above="@+id/textView2"
android:layout_toRightOf="@+id/imageView"
android:layout_toEndOf="@+id/imageView" />

<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:textAppearance="?android:attr/textAppearanceSmall"
```



```

android:text="Small Text"
android:id="@+id/textView2"
android:layout_toEndOf="@+id/imageView"
android:layout_alignBottom="@+id/imageView"
android:layout_toRightOf="@+id/imageView" />

<Space
android:layout_width="20dp"
android:layout_height="20dp"
android:layout_below="@+id/textView2"
android:id="@+id/space" />
</RelativeLayout>

```

Listing 3.1 Przykład budowy layout w języku XML.

3.3 Baza danych

W aplikacji Mobilny Program Edukacyjny do utworzenia bazy danych, oraz wykonywania do niej zapytań wykorzystano bibliotekę SugarORM [9]. Jest to biblioteka rozpowszechniana na licencji open-source, która umożliwia odwzorowanie klas języka Javy na bazę danych o relacyjnym charakterze. Wszelkie informacje dotyczące instalacji oraz konfiguracji biblioteki można znaleźć na stronie internetowej projektu [9].

3.4 Podstawowe pojęcia związane z programowaniem aplikacji na platformę Android

3.4.1 Aktywność

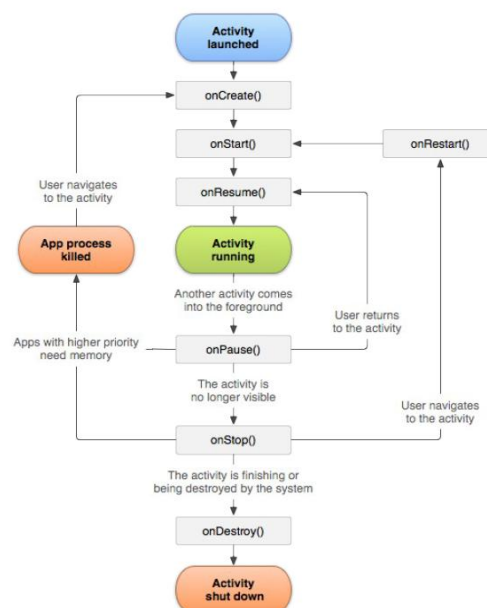
Aktywności stanowią główne elementy z jakich zbudowana jest aplikacja. Zadaniem aktywności jest realizowanie podstawowych operacji takich jak generowanie okna aplikacji, komunikacja z użytkownikiem, wywołanie innych aktywności. W kontekście języka Java, aktywność jest klasą, która jest dziedziczona w każdej aplikacji na platformę Android.

3.4.2 Cykl życia aktywności

Większość aplikacji zawiera kilka aktywności, a każda aktywność posiada swój własny cykl życia. Metody które należy zaimplementować w aktywności decydują o jej złożoności. Rysunek 3.2 ilustruje cykl życia aktywności. Do metod związanych z cyklem życia aktywności należą:

- `onCreate()` - tworzy nową instancję każdej aktywności. Implementacja tej metody następuje raz w całym cyklu życia aktywności. W niej należy inicjować pozostałe elementy aplikacji.

- `onRestart()` - metoda wywoływana w przypadku gdy aktywność została wcześniej wstrzymana, a teraz następuje do niej powrót.
- `onStart()` - metoda wywoływana jest gdy aktywność zaczyna być widoczna dla użytkownika.
- `onResume()` - metoda wywoływana gdy aktywność rozpoczyna interakcję z użytkownikiem.
- `onPause()` - metoda wywoływana gdy system rozpoczyna wznowienie poprzedniej aktywności.
- `onStop()` - metoda wywoływana gdy aktywność nie jest już widoczna dla użytkownika, ponieważ inna aktywność została wznowiona i ją przykrywa.
- `onDestroy()` - metoda wywoływana gdy aktywność jest niszczona. Metoda ta może wywołana zostać poprzez wykonanie metod `finish()` bądź przez system, aby oszczędzić pamięć.



Rysunek 3.2 Cykl życia aktywności [12].

3.4.3 Widoki

Widoki to obiekty tworzące interfejs użytkownika (UI). Dzięki nim użytkownik komunikuje się z aktywnościami. Językiem opisującym widoki jest język XML.

3.4.4 Kontekst aplikacji

Kontekst aplikacji służy do udostępniania specyficznych zasobów i klas, a także dostarcza informacji o środowisku aplikacji. W kodzie jest zaimplementowany za pomocą klasy `Context`. Jest ona abstrakcyjną klasą, której implementacja jest dostarczana wraz z systemem Android.

3.4.5 Zamiary

Każdorazowe utworzenie obiektu `Intent` wyraża zamiar wykonania pewnej operacji. Obiekt klasy `Intent` jest często stosowany jako argument metody `startActivity(Intent intent)`, w celu uruchomienia z góry ustalonej w argumentach konstruktora aktywności.

3.5 Implementacje modułów

3.5.1 Implementacja modułu Główny ekran aplikacji

Moduł *Główny ekran aplikacji*, jest pierwszą aktywnością jaką zobaczy użytkownik, zaraz po uruchomieniu aplikacji. Stanowi on swego rodzaju menu, w którym użytkownik po naciśnięciu wybranego przycisku przechodzi w inną aktywność. Listing 3.2 przedstawia mechanizm przejścia do innej aktywności. W tym przypadku do aktywności związanej z przedmiotami językowymi.

```
Button language = (Button) findViewById(R.id.language_button);
language.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent intent = new
            Intent(context, LanguageActivity.class);
        startActivity(intent);
    }
});
```

Listing 3.2 Mechanizm przejścia do aktywności *LanguageActivity.class*.

W przypadku przechodzenia do aktywności związanych z przedmiotami ścisłymi, oraz statystykami mechanizm jest identyczny jak pokazano na Listingu 3.2, z różnicą drugiego argumentu konstruktora klasy `Intent`. W przypadku przedmiotów ścisłych, ma postać `NumberActivity.class`, a dla statystyk `StatsActivity.class`.

Drugim zadaniem modułu *Główny ekran aplikacji* jest inicjalizacja bazy danych w przypadku, gdy aplikacja uruchamiana jest po raz pierwszy. Listing 3.3 pokazuje

mechanizm weryfikacji istnienia bazy danych. Na Listingu 3.4 pokazano warunek inicjalizacji bazy danych. Listing 3.5 przedstawia fragment jednej metody inicjalizującej bazę danych.

```
private boolean checkDataBase() {
    SQLiteDatabase checkDB = null;
    try {
        checkDB = SQLiteDatabase.openDatabase(
            getApplicationContext().getDatabasePath(
                DB_FULL_PATH).getPath(), null,
                SQLiteDatabase.OPEN_READONLY);
        checkDB.close();
    } catch (SQLiteException e) {
    }
    return checkDB != null ? true : false;
}
```

Listing 3.3 Metoda zwracająca wartość logiczną w zależności, czy istnieje już baza danych.

```
if(!checkDataBase()) {
    initOrthographyDb();
    initStatsDb();
    initHangmanDb();
    initPolishLettersDb();
    initEnglishLettersDb();
    initEnglishWordsDb();
    initElementsDb();
}
```

Listing 3.4 inicjalizacja bazy danych.

```
public void initElementsDb() {
    Element e = new
    Element("H", "wodór", "Hydrogen", 1, 1, 1, "gaz", "niemetal");
    e.save();
    e = new
    Element("He", "hel", "Helium", 2, 18, 1, "gaz", "gaz_szlachetny");
    e.save();
    e = new Element("Li", "lit", "Lithium", 3, 1, 2, "stały", "metal");
    e.save();
    e = new
    Element("Be", "beryl", "Beryllium", 4, 2, 2, "stały", "metal");
    e.save();
}
```

Listing 3.5 Fragment metody initElementsDb().

3.5.2 Implementacja modułu Programy przedmiotów językowych

Moduł Programy przedmiotów językowych jest aktywnością implementującą słuchacza onItemClickListener interfejsu AdapterView. Sama aktywność składa się z metody onCreate oraz przysłoniętej metody onItemClick. Listing 3.7 przedstawia implementację przysłoniętej metody onItemClick. Zadaniem tej

aktywności jest przy pomocy obiektu klasy `Intent` uruchomienie wybranej gry edukacyjnej dostępnej na liście. Lista ta jest reprezentowana standardowym komponentem `ListView`, na rzecz której wywoływany jest słuchacz. Ponieważ interfejs (`BaseAdapter`) w swojej standardowej implementacji nie przewiduje, że na liście znajdują się obiekty graficzne (np. ikony), należy zaimplementować własny adapter rozszerzający interfejs `BaseAdapter`. Listing 3.6 przedstawia implementację własnej klasy `MyAdapter`.

```
class SingleRow {
    String title;
    String desc;
    int image;

    SingleRow(String title, String desc, int image) {
        this.title = title;
        this.desc = desc;
        this.image = image;
    }
}

class MyAdapter extends BaseAdapter {
    ArrayList<SingleRow> list;
    Context context;
    MyAdapter(Context c) {
        list = new ArrayList<SingleRow>();
        Resources res = c.getResources();
        context = c;
        String [] titles = res.getStringArray(
            R.array.titles_language);
        String [] descS = res.getStringArray(
            R.array.descs_language);
        int [] images = {R.drawable.pioro,
                        R.drawable.wisielec,
                        R.drawable.wisielec_eng,
                        R.drawable.angielski};

        for(int i = 0; i < 4; i++) {
            list.add(new
SingleRow(titles[i],descs[i],images[i]));
        }
    }
    @Override
    public int getCount() {
        return list.size();
    }

    @Override
    public Object getItem(int i) {
        return list.get(i);
    }

    @Override
```

```

public long getItemId(int i) {
    return i;
}

@Override
public View getView(int i, View view, ViewGroup viewGroup)
{
    LayoutInflater inflater =
        (LayoutInflater) context.getSystemService(
            Context.LAYOUT_INFLATER_SERVICE);
    View row = inflater.inflate(
        R.layout.singel_row,viewGroup,false);
    TextView title = (TextView) row.findViewById(
        R.id.textView);
    TextView desc = (TextView)
        row.findViewById(R.id.textView2);
    ImageView image = (ImageView)
        row.findViewById(R.id.imageView);
    SingleRow temp = list.get(i);
    title.setText(temp.title);
    desc.setText(temp.desc);
    image.setImageResource(temp.image);

    return row;
}
}

```

Listing 3.6 Implementacja własnej klasy adaptera.

```

@Override
public void onItemClick(AdapterView<?> adapterView, View view,
int i, long l) {
    switch (i) {
        case 0:
            Intent intent =
                new Intent(getApplicationContext(),
                    OrthographyActivity.class);
            startActivity(intent);
            break;
        case 1:
            intent =
                new Intent(getApplicationContext(),
                    HangmanActivity.class);
            intent.putExtra("language","polish");
            startActivity(intent);
            break;
        case 2:
            intent =
                new Intent(getApplicationContext(),
                    HangmanActivity.class);
            intent.putExtra("language","english");
            startActivity(intent);
            break;
        case 3:
            intent =
                new Intent(getApplicationContext(),

```

```

                                EnglishActivity.class);
                                startActivity(intent);
                                break;
                            }
                        }
                    }
}

```

Listing 3.7 Implementacja przysłoniętej metody słuchacza.

Na Listingu 3.7 pokazano, że dla modułu *Wisielec z Polski*, oraz *Wisielec z Anglii* niezbędne jest wywołanie metody `putExtra` na rzecz obiektu klasy `Intent`, dzięki której, w zależności od jej drugiego argumentu, ta sama aktywność będzie funkcjonować na dwa różne sposoby.

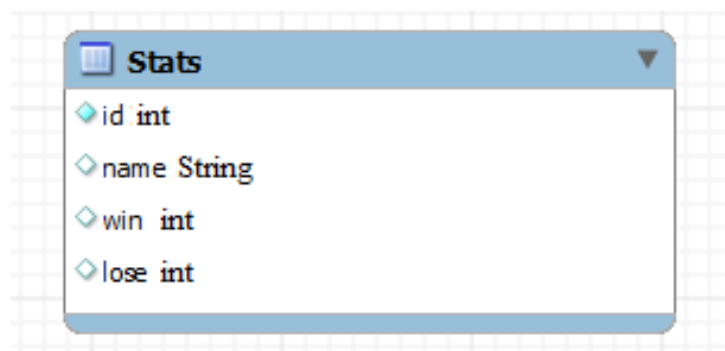
3.5.3 Implementacja modułu Programy przedmiotów ścisłych

Implementacja tego modułu działa na takiej samej zasadzie jak implementacja modułu *Programy przedmiotów językowych*. Różnice stanowią wywoływane aktywności w przypadku zdarzenia `onItemClick` oraz identyfikatory zasobów wczytywanych w konstruktorze klasy `MyAdapter`.

3.5.4 Implementacja modułu Statystyki

Implementacja tego modułu polega na odczytaniu wartości z tabeli `Stats` bazy danych, oraz przedstawienia tych danych za pomocą obiektu `list` klasy `ListView`. Rysunek 3.3. przedstawia graficzną reprezentację tabeli `Stats` w bazie danych. Ze względu na zmianę źródła danych wyświetlanych na liście, konieczne jest ponowne implementowanie własnego adaptera rozszerzającego klasę (`BaseAdapter`) dla obiektu `list`.

Listing 3.8 przedstawia wykorzystanie klasy `SugarRecord` do utworzenia tabeli w bazie danych. Listing 3.9 przedstawia fragmenty kodu modułu, przedstawiające mechanizm odczytu wartości z bazy danych za pomocą biblioteki `SugarORM`.



Rysunek 3.3 Graficzna reprezentacja tabeli Stats.

```

package com.zasady.sstarzak.mobilnyprogram;
import com.orm.SugarRecord;

public class Stats extends SugarRecord<Stats> {
    String name;
    int win;
    int lose;

    public Stats() {
    }

    public Stats(String name, int win, int lose) {
        this.name = name;
        this.win = win;
        this.lose = lose;
    }

    public void addNewStats(int new_win, int new_lose) {
        this.win += new_win;
        this.lose += new_lose;
    }
}

```

Listing 3.8 Wykorzystanie klasy SugarRecord do utworzenia tabeli w bazie danych.

```

List<Stats> stats = Stats.listAll(Stats.class);
public String [] getLosses(){
    String [] losses = new String[(int)
    Stats.count(Stats.class,null,null)];
    int i =0;
    for(Stats s : stats) {
        losses[i++] = String.valueOf(s.lose);
    }
    return losses;
}

```

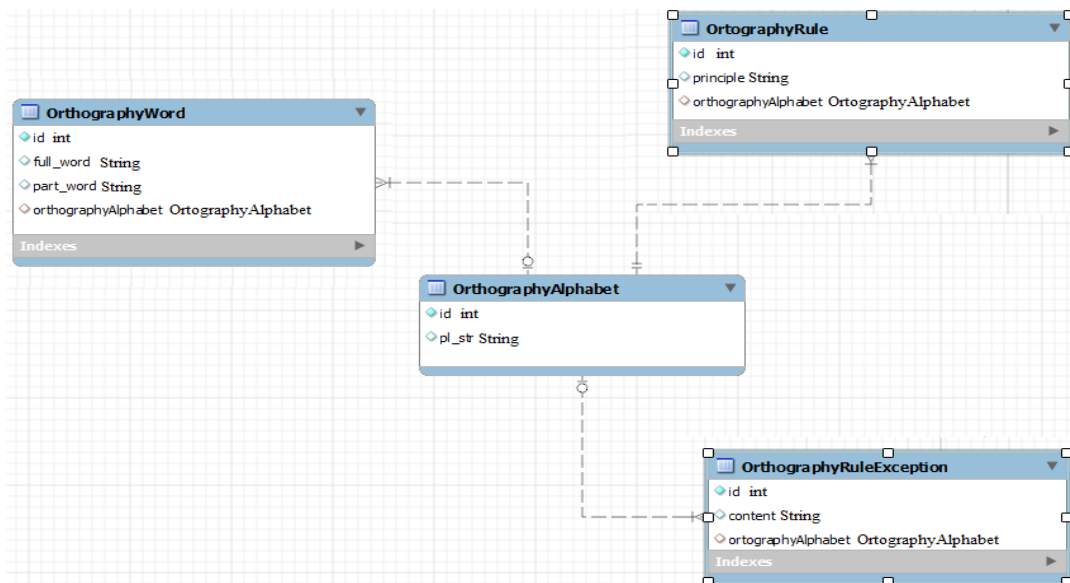
Listing 3.9 Mechanizm odczytu wartości z bazy danych.

3.5.5 Implementacja modułu Koniec

Zasada działania tego modułu polega na wywołaniu metody *finish*. Metoda ta wywołana z aktywności modułu *Główny ekran aplikacji* powoduje jej przejście do etapu *onDestroy* i tym samym spowoduje zamknięcie aplikacji.

3.5.6 Implementacja modułu Ortografia dla najlepszych

W momencie wywołania metody *onCreate* aktywności *OrtographyActivity* następuje inicjalizacja wszystkich pól tej klasy. Ustawiane są odpowiednie czcionki tekstowe. Wywoływane zostają słuchacze na rzecz przycisków, z których użytkownik będzie korzystać. Inicjalizowane zostają także wszystkie zmienne, których wartościami są dane odczytane z tabel bazy danych. Rysunek 3.4 przedstawia tabele bazy danych wykorzystanych w obrębie modułu.



Rysunek 3.4 Graficzna reprezentacja tabel wraz z ich relacjami.

Następnie wywoływana jest metoda `orthographyTest`. Na początku tej metody losowana jest para polskich znaków ortograficznych znajdujących się w tabeli `OrthographyAlphabet`. Kolejnym krokiem algorytmu jest wyszukanie wszystkich słów, które zawierają jeden z wylosowanej pary, polski znak ortograficzny. Następnie spośród wszystkich wyszukanych słów, wybierane jest losowo jedno słowo. Proces ten przedstawiono na Listingu 3.10.

```

do {
    randomvalueforid = (new Random()).nextInt((int)
(alphabetcount));
} while (randomvalueforid % 2 == 1);

letter =
OrthographyAlphabet.findById(OrthographyAlphabet.class,
firstletter.getId() + randomvalueforid);
letterpair =
OrthographyAlphabet.findById(OrthographyAlphabet.class,
firstletter.getId() + randomvalueforid + 1);

filteredwords.clear();

for (OrthographyWord w : orthographyWords) {
    if (w.orthographyAlphabet.pl_str.equals(letter.pl_str) ||
        w.orthographyAlphabet.pl_str.equals(letterpair.pl_str))
        filteredwords.add(w);
}
filteredwordscount = (long) filteredwords.size();
orthographyWord =
filteredwords.get((new Random()).nextInt((int)
filteredwordscount));

```

Listing 3.10 Losowanie jednego słowa.

Kolejnym krokiem algorytmu jest zastąpienie polskiego znaku ortograficznego trzema kropkami. W tym momencie program oczekuje na zdarzenie `onClick` wywołane przez użytkownika. Gdy to zdarzenie nastąpi, wykonana zostaje instrukcja warunkowa sprawdzająca, czy konkatencja trzech ciągów znaków jest identyczna z wylosowanym słowem. W zależności od tego czy warunek został spełniony czy nie, wywołana zostaje jedna metoda `onCorrectAnswer` bądź `onIncorrectAnswer`. Proces ten przedstawiono na Listingu 3.11.

```
String puzzle_word = orthographyWord.part_word.substring(0,
diff)
+ "...";
+ orthographyWord.part_word.substring(diff);

@Override
public void onClick(View view) {
    answer = orthographyWord.part_word.substring(0, diff) +
    letter.pl_str +

    orthographyWord.part_word.substring(diff);

    if (answer.equals(orthographyWord.full_word)) {
        onCorrectAnswer();
    } else {
        onIncorrectAnswer();
    }
}
```

Listing 3.11 Kroki algorytmu prowadzące do jednej z dwóch metod.

W zależności od wywołanej metody zostaje wywołana odpowiednia animacja na rzecz widoku pola tekstowego. Listing 3.12 przedstawia kod statycznej metody klasy `MyViewAnimations`, której wywołanie powoduje uruchomienie animacji podanego w parametrach obiektu klasy `View`.

```
public static void myNegativeScaleAnimation(final View view,
final long time) {
    ScaleAnimation sa = new ScaleAnimation(1, 1, 1, .01f);
    sa.setDuration(time);
    view.startAnimation(sa);
    sa.setAnimationListener(new Animation.AnimationListener() {
        @Override
        public void onAnimationStart(Animation animation) {

        }
        @Override
        public void onAnimationEnd(Animation animation) {
            ScaleAnimation sa = new ScaleAnimation(1, 1, 1, 0.01f,
```

```

1);
        sa.setDuration(time / 2);
        view.startAnimation(sa);
    }

    @Override
    public void onAnimationRepeat(Animation animation) {

    }

});
}

```

Listing 3.12 Animacja wywołana z metody `onIncorrectAnswer`.

Po zakończeniu animacji, ponownie zostaje wywołana metoda `orthographyTest`. Cały algorytm zostaje powtórzony. Na koniec działania aktywności, w metodzie `onStop` następuje aktualizacja pola tabeli `Stats` dotyczącego tej aktywności. Proces aktualizacji pola tabeli przedstawia Listing 3.13.

```

@Override
protected void onStop() {
    super.onStop();

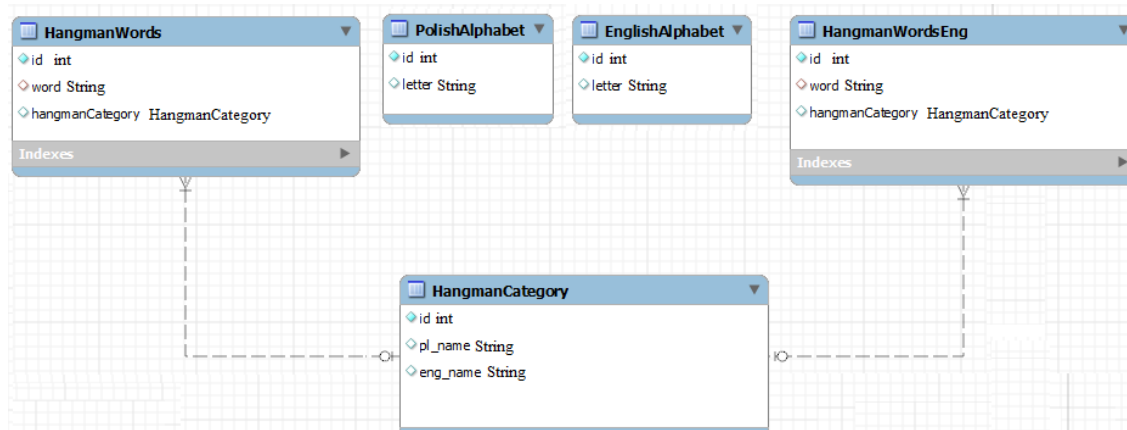
    Stats s = Stats.find(Stats.class, "name = ?",
        "Ortografia").get(0);
    s.addNewStats(wins, loses);
    s.save();
}

```

Listing 3.13 Aktualizacja pola tabeli.

3.5.7 Implementacja modułu Wisielec z Polski

W momencie wywołania metody `onCreate` aktywności `HangmanActivity`, następuje inicjalizacja wszystkich pól tej klasy. Wywoływane zostają słuchacze na rzecz przycisków, z których użytkownik będzie korzystać. Ustawiane są odpowiednie czcionki tekstowe. Inicjalizowane zostają także wszystkie zmienne, których wartościami są dane odczytane z tabel bazy danych. Rysunek 3.5 przedstawia tabele bazy danych wykorzystanych w obrębie modułu.



Rysunek 3.5 Graficzna reprezentacja tabel wraz z ich relacjami.

W kolejnym kroku wywołana zostaje metoda `hangmanTest`. Na początku tej metody następuje przypisanie do zmiennej całkowitej `attempts` wartości 5, a następnie losowane jest jedno słowo z tabeli `HangmanWords`. Na podstawie wylosowanego słowa odczytywana jest z tabeli `HangmanCategory` kategoria tematyczna wylosowanego słowa. Następnie zostaje wylosowany jeden znak należący do wylosowanego słowa. Kolejnym krokiem algorytmu jest wygenerowanie ciągu znaków, który ma tę samą długość co wylosowane słowo. Pod każdym indeksem tego ciągu znajduje się znak '-' za wyjątkiem tych indeksów, gdzie występuje wylosowany znak. Proces ten przedstawiono na Listingu 3.14.

```
randomletterindex = new
Random().nextInt(hangman_word.length());
for (int a = 0; a < hangman_word.length(); a++) {
    if (hangman_word.charAt(a) !=
hangman_word.charAt(randomletterindex))
        hangman_puzzle += "-";
    else
        hangman_puzzle +=
hangman_word.charAt(randomletterindex);
}
```

Listing 3.14 Generowanie ciągu znaków.

W tym momencie program oczekuje na zdarzenie `onClick` wywołane przez użytkownika. Gdy to zdarzenie nastąpi, wywołana zostaje metoda `hangmanGame`. Zadaniem tej metody jest weryfikacja, czy znak znajdujący się pod wybranym przyciskiem należy do wylosowanego słowa. Jeżeli tak, to następuje zamiana znaku '-' z wygenerowanego ciągu znaków na wybrany znak pod każdym indeksem jego występowania. W przeciwnym wypadku, następuje dekrementacja zmiennej `attempts`, oraz wczytywany jest nowy obraz dla obiektu `hangman_image` klasy `ImageView`. W

obu przypadkach wybrany przycisk jest ukrywany. Metoda `hangmanGame` wywoływana jest dopóki zmienna `attempts` jest różna od zera lub wygenerowany ciąg znaków jest równy wylosowanemu słowu. W pierwszym przypadku wykonana zostaje metoda `onIncorrectAnswer`, w drugim przypadku `onCorrectAnswer`. Proces ten przedstawiono na Listingu 3.15.

```
public void hangmanGame(String selected_letter) {
    String hangman_temp = "";
    if (hangman_word.contains(selected_letter)) {
        for (int a = 0; a < hangman_word.length(); a++) {
            if (hangman_word.charAt(a) ==
                selected_letter.charAt(0) &&
                hangman_puzzle.charAt(a) == '-')
                hangman_temp += hangman_word.charAt(a);
            else
                hangman_temp += hangman_puzzle.charAt(a);
        }
        hangman_puzzle = hangman_temp;

        MyViewAnimations.myScaleAnimation(tv_hangman, 25);

        tv_hangman.setText(hangman_puzzle);
    } else {
        attempts--;
        MyViewAnimations.myHangmanShakerAnimation(
            hangman_image, 10, 5);
        switch (attempts) {
            case 4:
                hangman_image.setImageResource(R.drawable.hangman2);
                break;
            case 3:
                hangman_image.setImageResource(R.drawable.hangman3);
                break;
            case 2:
                hangman_image.setImageResource(R.drawable.hangman4);
                break;
            case 1:
                hangman_image.setImageResource(R.drawable.hangman5);
                break;
            case 0: {
                hangman_image.setImageResource(R.drawable.hangman6);
                onIncorrectAnswer();
                break;
            }
        }
    }
}
if (hangman_word.equals(hangman_puzzle)) {
    onCorrectAnswer();
}
}
```

Listing 3.15 Proces generowania ciągu znaków.

W zależności od wywołanej metody zostaje wywołana odpowiednia animacja na rzecz widoku pola tekstowego oraz na rzecz obiektu `ImageView`. Na koniec działania aktywności, w metodzie `onStop` następuje aktualizacja pola tabeli `Stats` dotyczącego tej aktywności.

3.5.8 Implementacja modułu Wisielec z Anglii

Moduł ten implementuję tę sama aktywność, co moduł *Wisielec z Polski*. Różnica polega na przekazywanym parametrze metody `putExtra` obiektu klasy `Intent` wywoływanym z poziomu modułu *Programy przedmiotów językowych*. Listing 3.16 przedstawia fragment instrukcji wyboru decydującej o parametrach wywoływanej aktywności. W aktywności modułów *Wisielca*, sprawdzana jest wartość odebranego parametru `"language"` przesłanego z poprzedniej aktywności. Proces ten przedstawia Listing 3.17.

```
case 1:
    intent = new Intent(getApplicationContext(),
        HangmanActivity.class);
    intent.putExtra("language", "polish");
    startActivity(intent);
    break;
case 2:
    intent = new Intent(getApplicationContext(),
        HangmanActivity.class);
    intent.putExtra("language", "english");
    startActivity(intent);
    break;
```

Listing 3.16 Wywołanie aktywności z parametrem.

```
intent_language = getIntent();
selected_language =
    intent_language.getStringExtra("language");

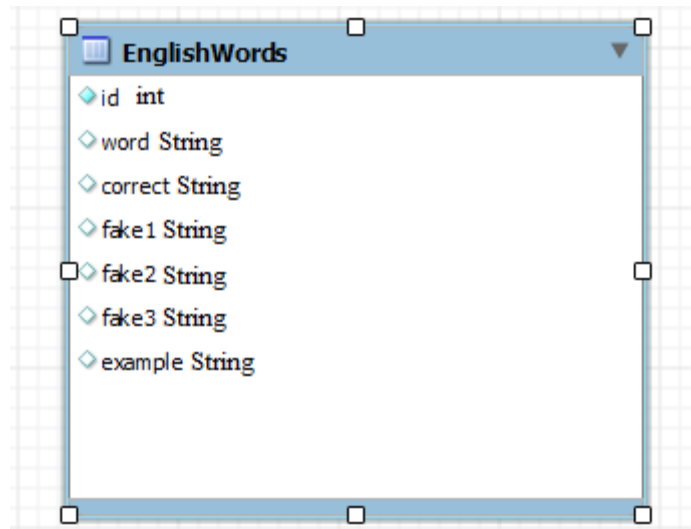
if (selected_language.equals("polish"))
    tv_category.setText("Kategoria: " +
        hangman_words.get((int)
            randomvalueforid).hangmanCategory.plname);
else if (selected_language.equals("english"))
    tv_category.setText("Category: " +
        hangman_english_words.get((int)
            randomvalueforid).hangmanCategory.engname);
```

Listing 3.17 Instrukcje wykonywane w zależności od parametru `language`.

Rezultatem opisanych działań jest korzystanie z innych tabel bazy danych. Odczytane słowo losowane jest z tabeli `HangmanWordsEng`, a etykiety przycisków są wartościami tabeli `EnglishAlphabet`.

3.5.9 Implementacja modułu Nauka angielskiego

W momencie wywołania metody `onCreate` aktywności `EnglishActivity` następuje inicjalizacja wszystkich pól tej klasy. Wywoływane zostają słuchacze na rzecz przycisków, z których użytkownik będzie korzystać. Ustawiane są odpowiednie czcionki tekstowe. Inicjalizowane zostają także wszystkie zmienne, których wartościami są dane odczytane z tabel bazy danych. Rysunek 3.6 przedstawia tabele bazy danych wykorzystanych w obrębie modułu *Nauka angielskiego*.



Rysunek 3.6 Graficzna reprezentacja tabeli.

Kolejnym krokiem jest odczytanie jednego rekordu o losowym identyfikatorze z tabeli `EnglishWords`. Następnie wylosowana zostaje liczba z przedziału od 0 do 3 która określa na którym obiekcie klasy `RadioButon` w polu `text` znajdować się będzie wartość pola `correct` tabeli `EnglishWords`. Dla pozostałych obiektów klasy `RadioButton` pola `text` otrzymują kolejno wartości pól `fake1`, `fake2`, `fake3` tabeli `EnglishWords`. Listing 3.18 przedstawia metodę `englishWordsTest` realizującą opisane kroki.

```
public void englishWordsTest() {
    random_value_for_id =
        (new Random()).nextInt((int) (english_words_count));
    english_word = EnglishWords.findById(
        EnglishWords.class, english_first_word.getId() +
        random_value_for_id);
    random_position_of_correct = new Random().nextInt(4);
    user_answer = -1;

    RadioGroup rg = (RadioGroup)
        findViewById(R.id.radioGroupEng);
    rg.clearCheck();
}
```



```

enableBotons(true);
switch (random_position_of_correct) {
    case 0: rb1.setText(english_word.correct);
            rb2.setText(english_word.fake1);
            rb3.setText(english_word.fake2);
            rb4.setText(english_word.fake3);break;
    case 1: rb2.setText(english_word.correct);
            rb1.setText(english_word.fake1);
            rb3.setText(english_word.fake2);
            rb4.setText(english_word.fake3);break;
    case 2: rb3.setText(english_word.correct);
            rb2.setText(english_word.fake1);
            rb1.setText(english_word.fake2);
            rb4.setText(english_word.fake3);break;
    case 3: rb4.setText(english_word.correct);
            rb2.setText(english_word.fake1);
            rb3.setText(english_word.fake2);
            rb1.setText(english_word.fake3);break;
}
tv_example.setText("Przykład: " + english_word.example);
tv_word.setText(english_word.word);
}

```

Listing 3.18 Metoda englishWordsTest.

W zależności od pola id widoku, w metodzie słuchacza podejmowana jest odpowiednia akcja. Przedstawia to Listing 3.19.

```

@Override
public void onClick(View view) {
    switch(view.getId()) {
        case R.id.english_radio1: user_answer = 0; break;
        case R.id.english_radio2: user_answer = 1; break;
        case R.id.english_radio3: user_answer = 2; break;
        case R.id.english_radio4: user_answer = 3; break;
        case R.id.english_words_check_button:
            englishWordGame(); break;
    }
}
}

```

Listing 3.19 Instrukcja wyboru zaimplementowana dla metody onClick.

Ostatnim krokiem algorytmu jest weryfikacja, czy wartość *id* wybranego obiektu klasy `RadioButton` jest taka sama jak *id* obiektu klasy `RadioButton`, pod którym znajduje się wartość pola `correct` tabeli `EnglishWords`. Krok ten przedstawia Listing 3.20.

```

public void englishWordGame() {
    enableBotons(false);
    if(random_position_of_correct == user_answer){
        final Handler h = new Handler();
        final Runnable r1 = new Runnable() {
            @Override

```

```

        public void run() {
            rl.setBackgroundResource(R.drawable.zeszyt);
            MyViewAnimations.myEnglishAnimation(rl, 500);
            englishWordsTest();
        }
    };
    h.postDelayed(r1, 2000);
    rl.setBackgroundResource(R.drawable.zeszyt_yes);
    wins++;

} else {
    final Handler h = new Handler();
    final Runnable r1 = new Runnable() {
        @Override
        public void run() {
            rl.setBackgroundResource(R.drawable.zeszyt);
            MyViewAnimations.myEnglishAnimation(rl, 500);
            englishWordsTest();
        }
    };
    h.postDelayed(r1, 2000);
    rl.setBackgroundResource(R.drawable.zeszyt_no);
    losts++;
}
}
}

```

Listing 3.20 Metoda weryfikująca.

3.5.10 Implementacja modułu Szybkie operacje

W momencie wywołania metody `onCreate` aktywności `FastOperationActivity` następuje inicjalizacja wszystkich pól tej klasy. Wywoływane zostają słuchacze na rzecz przycisków, z których użytkownik będzie korzystać. Ustawiane są odpowiednie czcionki tekstowe. Kolejnym krokiem jest utworzenie obiektu klasy `CountDownTimer` oraz wywołanie na rzecz tego obiektu metody `start`. Operację tę przedstawia Listing 3.21.

```

cdt = new CountDownTimer(cdt_time, 100) {
    @Override
    public void onTick(long l) {
        countdown_tv.setText(String.valueOf(l / 1000));
        if(l / 1000 <= 5) {
            MyViewAnimations.myTimerShakeAnimation(countdown_tv, 550);
            countdown_tv.setTextColor(Color.RED);
        }
    }
    @Override
    public void onFinish() {
        countdown_tv.setText("0");
        onIncorrectAnswer();
    }
}.start();

```

```

cdt2 = new CountDownTimer(cdt_time, 900) {
    @Override
    public void onTick(long l) {
        switch (attempts) {
            case 2: MyViewAnimations.myJumpingAnimation(
                operator1_tv, 900, operator1_tv.getHeight() / 2);
            break;
            case 1: MyViewAnimations.myJumpingAnimation(
                operator2_tv, 900, operator2_tv.getHeight() / 2);
            break;
        }
    }
    @Override
    public void onFinish() {

    }
}.start();

```

Listing 3.21 Implementacja obiektu klasy CountDownTimer.

Kolejnym krokiem jest wygenerowanie równania matematycznego, przy losowo wybranych liczbach dla operatorów. Procedurę tę przedstawia Listing nr 3.22

```

number1 = r.nextInt(2 * max_number) - 15;
number2 = r.nextInt(max_number) + 1;
number3 = r.nextInt(max_number) + 1;

operator1 = r.nextInt(operators_count);
operator2 = r.nextInt(operators_count);

```

Listing 3.22 Generowanie liczb i operatorów równania.

Po dwóch zdarzeniach onClick wywołanych przez użytkownika, rusza do pracy fastOperationTest, której zadaniem jest sprawdzenie równości wygenerowanego równania z równaniem utworzonym przez użytkownika. Listing 3.23 przedstawia metodę fastOperationTest. W implementacji tej metody użyta została otwarta biblioteka JEXL (Java Expression Language) [10].

```

public void fastOperationTest(String sOperator1, String
sOperator2) {

    JexlEngine jexl = new JexlEngine();

    Expression func =
jexl.createExpression("x1" + sOperator1 +
"x2" + sOperator2 +
"x3");

    MapContext mc = new MapContext();

    mc.set("x1", number1);

```

```

mc.set("x2", number2);
mc.set("x3", number3);

answer_number = Integer.parseInt(
func.evaluate(mc).toString());

if (answer_number.equals(correct_number))
{
    onCorrectAnswer();
}
else
{
    onIncorrectAnswer();
}
}

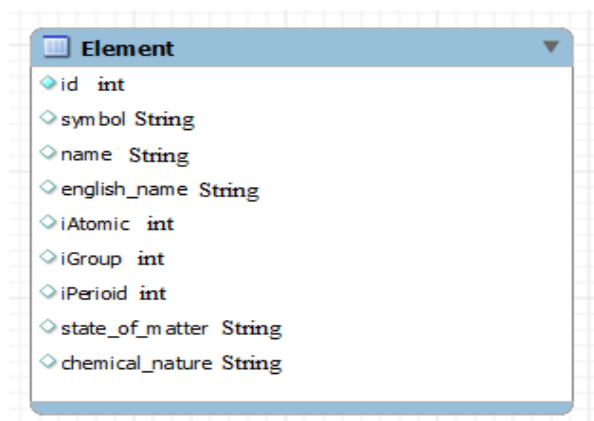
```

Listing 3.23 Metoda fastOperationTest.

3.5.11 Implementacja modułu Tablica Mendelejewa

W momencie wywołania metody onCreate aktywności PeriodicTableActivity następuje inicjalizacja wszystkich pól tej klasy. Wywoływane zostają słuchacze na rzecz przycisków, z których użytkownik będzie korzystać. Ustawiane są odpowiednie czcionki tekstowe. Inicjalizowane zostają także wszystkie zmienne, których wartościami są dane odczytane z tabel bazy danych. Rysunek 3.7 przedstawia tabelę Element bazy danych wykorzystanej w obrębie modułu.

Aby możliwe było ułożenie obiektów klasy Button tak, aby graficznie przypominały tablice Mendelejewa, należało zaimplementować metodę, która w zależności od id obiektu klasy Button, umieszczała go w obiekcie typu TableRow, który następnie umieszczany został w obiekcie TableLayout. Listing 3.24 przedstawia fragmenty kodu realizujące kroki tego algorytmu.



Element	
id	int
symbol	String
name	String
english_name	String
iAtomic	int
iGroup	int
iPeriod	int
state_of_matter	String
chemical_nature	String

Rysunek 3.7 Graficzna reprezentacja tabeli.

```

for (int a = 0; a < 10 * 18; a++) {
    if (a % 18 == 0) {
        periodic_tr = new TableRow(this);
        periodic_tl.addView(periodic_tr);
    }
    GradientDrawable gd = new GradientDrawable();
    gd.setCornerRadius(15);
    gd.setStroke(1, 0xFF000000);

    Button button = new Button(this);
    button.setId(a * 0x64);
    button.setTextSize(TypedValue.COMPLEX_UNIT_SP, 6.5f);

    button.setTypeface(tf);
    button.setOnClickListener(this);
    if (checkGroupAndPerioidForElement(a)) {
        button.setText(elements.get(index).symbol);
        gd.setColor(colorElement(elements.get(index)));
        button.setBackground(gd);
        index++;
    } else {
        button.setVisibility(View.INVISIBLE);
    }

    periodic_tr.addView(button, tr_params);
}
public boolean checkGroupAndPerioidForElement(int value) {
    if (value >= 1 && value <= 16 ||
        value >= 20 && value <= 29 ||
        value >= 38 && value <= 47 ||
        value >= 126 && value <= 144 ||
        value >= 144 && value <= 146 ||
        value >= 162 && value <= 164 ||
        value == 92 ||
        value == 110)
        return false;
    else
        return true;
}

```

Listing 3.24 Metoda checkGroupAndPerioidForElement oraz jej wywołanie.

Kolorowanie obiektów klasy Button realizuje metoda colorElement, która wywoływana jest gdy metoda checkGroupAndPerioidForElement zwróci wartość logiczną true. Przedstawia to Listing 3.25.

```

public int colorElement(Element e) {
    if (e.chemical_nature.equals("metal"))
        return Color.rgb(175, 238, 238);
    else if (e.chemical_nature.equals("niemetal"))
        return Color.rgb(221, 160, 221);
    else if (e.chemical_nature.equals("gaz_szlachetny"))
        return Color.YELLOW;
    else if (e.chemical_nature.equals("metal/półmetal"))
        return Color.rgb(250, 128, 114);
    else if (e.chemical_nature.equals("niemetal/półmetal"))
        return Color.rgb(154, 205, 50);
    else if (e.chemical_nature.equals("prawdopodobnie_niemetal"))

```

```

        return Color.WHITE;

    return 0;
}

```

Listing 3.25 Metoda colorElement.

3.5.12 Implementacja modułu Tablica Mendelejewa ENG

Moduł ten implementuję tę sama aktywność, co moduł Tablica Mendelejewa. Różnica polega na przekazywanym parametrze metody `putExtra` obiektu klasy `Intent` wywoływanym z poziomu modułu *Programy przedmiotów ścisłych*. Listing 3.26 przedstawia fragment instrukcji wyboru decydującej o parametrach wywoływanej aktywności. W aktywności modułów Tablica Mendelejewa, sprawdzana jest wartość odebranego parametru `"language"` przesłanego z poprzedniej aktywności. Proces ten przedstawia Listing 3.27.

```

case 1:
    intent = new Intent(getApplicationContext(),
        PeriodicTableActivity.class);
    intent.putExtra("language", "polish");
    startActivity(intent);
    break;
case 2:
    intent = new Intent(getApplicationContext(),
        PeriodicTableActivity.class);
    intent.putExtra("language", "english");
    startActivity(intent);
    break;

```

Listing 3.26 Wywołanie aktywności z dodatkowym parametrem

```

if (language.equals("polish")) {
    element_tv.setText(elements.get((int)
        randomValueForId).name);
} else if (language.equals("english")) {
    element_tv.setText(elements.get((int)
        randomValueForId).english_name);
}

```

Listing 3.27 Instrukcje wykonywane w zależności od parametru language.

Rezultatem opisanych działań jest korzystanie z innych pól tabeli `Elements` bazy danych. W przypadku modułu *Tablicy Mendelejewa ENG* pole to ma nazwę `english_name`.

3.5.13 Implementacja modułu Systemy liczbowe

W momencie wywołania metody `onCreate` aktywności `NumeralSystemActivity` następuje inicjalizacja wszystkich pól tej klasy.

Wywoływane zostają słuchacze na rzecz przycisków, z których użytkownik będzie korzystać. Ustawiane są odpowiednie czcionki tekstowe. Kolejnym krokiem jest wywołanie metody `numeralSystemGame`. Celem tej metody jest utworzenie obiektu klasy `CountDownTimer`, oraz ustawienie wartości pola `text` obiektów klasy `RadioButton`. Fragment kodu metody `numeralSystemGame` przedstawiono na Listingu 3.28.

```
correct = new Random().nextInt(max_random_range) +
min_random_value;

corrects = new String[3];

corrects[0] = correct.toString();
corrects[1] = Integer.toHexString(correct);
corrects[2] = Integer.toBinaryString(correct);

for (int a = 0; a < 3; a++) {
    RadioButton rb =
        (RadioButton) findViewById(id_for_correct[a]);
    rb.setText(corrects[a]);
}

Integer[] inc = generateInncorrectNumbers(correct);

for (int a = 0; a < 9; a++) {
    RadioButton rb = (RadioButton) findViewById(radioIds[a]);

    if (!Arrays.asList(id_for_correct).contains(rb.getId())) {
        if (a <= 2)
            rb.setText(Integer.toString(inc[a]));
        if (a > 2 && a <= 5)
            rb.setText(Integer.toHexString(inc[a]));
        if (a > 5)
            rb.setText(Integer.toBinaryString(inc[a]));
    }
}
```

Listing 3.28 Fragment metody `numeralSystemGame`.

```
public Integer[] generateInncorrectNumbers(int correct) {
    Integer[] inc = new Integer[9];
    int index = 0;
    int random;

    do {
        do {
            random = new Random().nextInt(max_random_range) +
                min_random_value;
        }
        while ((random == correct) ||
            Arrays.asList(inc).contains(random));
    }
```



```
        inc[index++] = random;
    }
    while (index < 9);
    return inc;
}
```

Listing 3.29 Metoda generateInncorrectNumbers.

Listing 3.29 przedstawia metodę `generateInncorrectNumbers`, której zadaniem jest wygenerowanie dziewięciu liczb różniących się od przyjmowanego parametru metody i między sobą.

Wnioski

Obecnie wśród programów edukacyjnych przeznaczonych na urządzenia mobilne można znaleźć kilka pozycji o zbliżonej tematyce do stworzonej aplikacji. Jedną z nich jest aplikacja Quizwanie stworzona na platformę Android, która ze względu na swoją popularność doczekała się także implementacji na systemy Windows Phone oraz iOS. Aplikacja ta dostarcza użytkownikom rozrywki intelektualnej, polegającej na przedstawieniu przez jednego z graczy pytań z wybranej kategorii tematycznej w taki sposób, by na podstawie dostarczonych wraz z pytaniem informacji oraz wiedzy ogólnej, drugi gracz był w stanie wydedukować prawidłową odpowiedź. Kolejną podobną aplikacją jest Kujon. Przeznaczona jest głównie dla uczniów szkoły podstawowej. Jej funkcjonalność oferuje naukę w zakresie języka polskiego (ortografię i fonetykę) oraz matematyki (podstawowe działania arytmetyczne). Innym przykładem jest stworzona na platformę Windows Phone aplikacja Pierwiastki. Jej głównymi odbiorcami są uczniowie szkół średnich o profilach chemicznych i ci wszyscy, którzy potrzebują informacji na temat konkretnego pierwiastka chemicznego. Warto również wspomnieć o programie Fizyka na 5. Jest to łatwa w obsłudze, darmowa aplikacja, która zawiera większość wzorów fizycznych wraz z opisami oraz ilustracjami obrazującymi zjawiska fizyczne. Jest idealna dla uczniów oraz studentów fizyki.

Celem pracy było napisanie mobilnego programu edukacyjnego w języku Java, dzięki któremu użytkownik mógłby poszerzać swoją wiedzę z różnych dziedzin naukowych. Aplikacja zawiera zestaw modułów edukacyjnych, a każdy z nich przynależy do jednej z dwóch kategorii tematycznych: języki bądź przedmioty ścisłe. Każdy moduł edukacyjny jest interaktywną grą, w której użytkownik ma wybrać poprawną odpowiedź. W zależności od poprawności wybranej odpowiedzi, użytkownik otrzymuje stosowny komunikat tekstowy lub komunikat animowany o tym, czy odpowiedział poprawnie, a jeśli nie, to która odpowiedź była poprawna. Użytkownik z poziomu głównego menu ma dostęp do statystyk. Statystyki przedstawiają nazwę modułu, liczbę poprawnych i niepoprawnych odpowiedzi, a także procentowy stosunek udzielonych odpowiedzi do rozegranych partii.

Głównym elementem nowości wprowadzonym do pracy jest wielotematyczność modułów edukacyjnych w stosunku do najbardziej popularnych aplikacji o tej samej

tematyce dostępnych w chwili obecnej na rynku. Moduły te dotyczą zagadnień języka polskiego, języka angielskiego, matematyki, chemii oraz informatyki.

Podczas realizacji autor napotkał kilka problemów. Pierwszym z nich było znalezienie odpowiednich narzędzi do budowy tabel w bazie danych. Rozwiązaniem okazało się wykorzystanie otwartej biblioteki SugarORM. Kolejnym problemem było rozwiązywanie równania matematycznego uwzględniając priorytety operatorów w module *Szybkie operacje*. Autor pracy wykorzystał w tym wypadku otwartą bibliotekę JEXL do budowania wyrażeń matematycznych..

Praca napisana jest w formie pozwalającej na jej rozbudowę. Dla każdej kategorii tematycznej możliwe jest dodanie nowych modułów edukacyjnych. Możliwym jest także dodanie odpowiedniej funkcjonalności, służącej do dołączenia przez użytkownika nowych rekordów do tabeli bazy danych, bądź pobierania tych rekordów z sieci Internet.

Literatura

- [1] Google Play: Quizwanie, <https://play.google.com/store/apps/details?id=se.feomedia.quizkampen.pl.lite>
- [2] Google Play: Kujon, <https://play.google.com/store/apps/details?id=mazakit.kujon>
- [3] Windows Phone: Pierwiastki, <http://www.windowsphone.com/pl-pl/store/app/pierwiastki/b5680c0d-abe7-4e08-a00c-556c49135e91>
- [4] Google Play: Fizyka na 5, <https://play.google.com/store/apps/details?id=Gecko.Droid.PhysicsHelper&hl=pl>
- [5] Geckonization, Pocket Physics, <http://geckonization.com/>
- [6] Cay S. Horstmann, Gary Cornell, “Java. Podstawy. Wydanie IX”, Wydawnictwo Helion, Warszawa 2013
- [7] Joshua Bloch, “Java. Efektywne programowanie. Wydanie II”, Wydawnictwo Helion, Warszawa 2009
- [8] Elizabeth Castro, “Po prostu XML”, Wydawnictwo Helion, Gliwice 2001
- [9] SugarORM, Overview, <http://satyan.github.io/sugar/>
- [10] Apache Commons, commons-jexl, <http://commons.apache.org/proper/commons-jexl/>
- [11] Developer Android, Android Studio, <http://developer.android.com/sdk/index.html>
- [12] Developer Android, Activity, <http://developer.android.com/reference/android/app/Activity.html>

Wykaz załączników

Płyta CD zawiera:

- Tekst pracy w formie dokumentu PDF
- Kod źródłowy aplikacji w katalogu: MobilnyProgramEdukacyjny
- Aplikację w katalogu: Aplikacja

Streszczenie

Sygnatura:

POLITECHNIKA RZESZOWSKA im. I. Łukasiewicza Rzeszów, 2015
Wydział Elektrotechniki i Informatyki
Katedra Podstaw Elektroniki

**STRESZCZENIE PRACY DYPLOMOWEJ INŻYNIERSKIEJ
(MOBILNY PROGRAM EDUKACYJNY)**

Autor: Szymon Starzak, nr albumu: EF-DI-127261

Opiekun: dr inż. Maciej Kusy

Słowa kluczowe: edukacyjny program, programowanie, mobilny program, android

Autor pracy zaprojektował oraz zaimplementował aplikację mobilną Mobilny Program Edukacyjny, dzięki której użytkownik może poszerzać swoją wiedzę z zakresu przedmiotów językowych oraz przedmiotów ścisłych. Aplikację stworzono w oparciu o technologię Java z wykorzystaniem środowiska Android Studio, platformy Android, frameworka SugarORM, oraz języka XML. Aplikacja umożliwia także, prowadzenie i przeglądanie statystyk dla każdego modułu edukacyjnego. W pracy dokonano również charakterystyki porównawczej podobnych aplikacji dostępnych na rynku.

RZESZOW UNIVERSITY OF TECHNOLOGY
Faculty of Electrical and Computer Engineering
Department of Electronics Fundamentals

Rzeszow, 2015

**DIPLOMA THESIS (BS) ABSTRACT
(MOBILE EDUCATIONAL PROGRAM)**

Author: Szymon Starzak, code: EF-DI-127261

Supervisor: Maciej Kusy, PhD, Eng.

Key words: educational program, programming, mobile program, android,

Author of the Diploma Thesis designed and implemented the mobile application called Mobilny Program Edukacyjny, which can be used to extend his knowledge in language and science domains. Application was created in Java programming language using Android Studio enviroment, Android platform, SugarORM framework, and XML language. Application also allowed the user to make and explore statistics for every single subject module. In this work, the comprative analysis of similar programs available in the market was performed.